DATA TRANSMISSION USING ATM OVER HYBRID FIBER COAX

BACKGROUND OF THE INVENTION

The invention pertains to the field of provision of ATM over SCDMA connectivity over a hybrid fiber coax plant between processes coupled to customer premises cable modems and head end (referred to herein as the CU or Central Unit) cable modems using an ATM cell with ATM header and using code division multiple access in at least the upstream direction.

In order to provide bidirectional digital data communication over a cable TV coaxial network to multiple subscribers with multiple services available over a single coax cable (hereafter called interactive systems), several problems have to be solved. First, there is the problem of noise and interference. A second major problem, but related to the first problem, is synchronization of data transmission so that effective, error-free communication can be achieved.

Further, because the subscribers are at physically different distances from the head end, the signals from each subscriber's modem arrive at the CU at different times because of different propagation delays. Because digital data is transmitted in frames and because all subscribers must be synchronized to the same frame timing, these different propagation delays for each subscriber cause problems in synchronizing data.

All of the above applies to the physical layer of the OSI model for data interchange between computers. At the higher protocol levels in the OSI model there are several standard protocols that are currently known. One of these protocols is the TCP/IP protocol used on the internet. This protocol is not satisfactory for provision of high demand services such as video teleconferencing and video on demand since TCP/IP has no provision to guarantee quality of service and provide guaranteed bandwidth capacity. Because there is no concept of reservation of bandwidth in TCP/IP protocol, it is not suitable for delivery simultaneously of audio, video and data services to multiple subscribers.

The ATM protocol is the currently favored local area network protocol which is designed to simultaneously deliver integrated voice, video and data services. However, the ATM protocol was designed for local area networks where there is no shared media which is used to simultaneously deliver ATM cells between more than one pair of communicating devices. ATM is a point to point communication protocol that cannot be

directly used on a CATV plant with its point to multipoint/multipoint to point topology.

Therefore, a need has arisen for a method of using ATM protocols to support interactive digital systems carried out over HFC cable plants.

SUMMARY OF THE INVENTION

Although a large part of this disclosure deals with the complexity of the physical layer for performing synchronous code division multiple access in at least the upstream direction thereby enabling the establishment of virtual links appearing logically to be point to point connections between the CU and the various RU cable modems, the technology to send ATM cells over a shared media like HFC is primarily discussed in the section below entitled "OPTIMIZED ATM OVER SCDMA". The fundamental idea is to convert a shared media such as a hybrid fiber coax cable TV system into a plurality of virtual links that appear logically to be separate point-to-point links between each RU and the CU and dedicated to only carrying data between that RU and the CU. Any technology can be used to establish the virtual links such as SCDMA, TDMA, FDMA or any other known multiplexing technology. Once the virtual links are established, ATM cells are packaged at the CU for transmission to the RUs over their assigned virtual links from raw data arriving from various local area or wide area networks. These ATM cells have virtual link headers added to their ATM cell headers that identify the RUs to which each cell is addressed, and each ATM cell is encoded with a start code by which the RU to which the cell is addressed can find the beginning and end of the cell. Likewise, the RU packages ATM cells from raw data arriving on local area or wide area networks or from local peripherals connect directly to the RU and encodes each ATM cell with a start code. The ATM cell is then transmitted upstream. No virtual link headers are added, because, in the SCDMA embodiments, the ATM cell is transmitted byte-by-byte in timeslots of a TDMA stream that feeds the SCDMA spectrum spreading circuitry that are assigned to only one RU. The CU reassembles this TDM stream and reassembles the ATM cells from the various timeslots. The timeslots in which the various pieces of each ATM cell arrived tells the CU from which RU the ATM cell originated.

In alternative embodiments, ATM cells are formed with reduced header size to improve bandwidth utilization efficiency, and unique encapsulation techniques are used to allow ATM cells to be generated from TCP/IP or Ethernet packets received at either the RU or CU, transmitted over virtual links and then disassembled into TCP/IP or Ethernet packets for transmission to a device over a local area network or out on a wide area network.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1A shows a typical data structure for a frame.

Figure 1B is a symbolic diagram illustrating the concepts involved in alignment to achieve frame synchronization, also called ranging herein, for the preferred species within the genus of the invention.

Figure 2, which is comprised of Fig. 2A, 2B, and 2C, is a flow chart for the general alignment process which is used in ranging processes carried out in all remote units (RUs) to set their delay vectors properly so as to be in alignment within the same frame.

Figure 3 is a general block diagram of the preferred embodiment of the transceiver circuitry included in each RU and CU.

Figure 4 is a diagram which helps illustrate the manner in which framer memory 300 is emptied for transmission.

Figure 5 maps each of 16 possible input points, i.e., permutations of the 4 bit "chips" in each symbol array, to a point in space defined by the in-phase or I axis for the real part and the quadrature or Q axis for the imaginary part of each point to implements M-ary QAM modulation.

Figure 6 is a table listing all the possible input points of Figure 5, I.e., the 16 combinations of 4 bit chips in the Code column and the corresponding 2's complement digital representation of the I and Q coordinates for each combination in the Inphase and Quadrature columns, respectively.

Figure 7A illustrates how the information vector [b] for each symbol has its energy spread over time by the process of code division multiplexing implemented using matrix multiplication of the information vector [b] of each symbol times a matrix of orthogonal codes.

Figure 7B is another illustration of the matrix multiplication process carried out in an orthogonal multiplexer 408 in Figure 3 to encode the real or I coordinates of each information vector using an orthogonal code matrix to generate the real or I coordinates of a result vector for use by the QAM modulator.

Figure 8 is a block diagram illustrating more details of the components and operation of the multiplexer 408 and the QAM modulator 410 used in the preferred species within the inventive genus.

Figure 9 is a plot of the changes in amplitude over time of the real components of the results vector for the array 409 illustrating the need for bandwidth limiting filters.

Figure 10 is a more detailed block diagram of the structure of the demodulator in the receive channel.

Figure 11 is a block diagram of the preferred embodiment of a transmitter

within the inventive genus of the invention using bit parsing from each timeslot, TDMA/CDMA spreading, M-ary QAM modulation, code diversity, encoding of each tribit with redundant bits for forward error correction and to aid Viterbi Decoding in the receiver, scrambling of bits of each tribit for security and signal to noise improvements, ranging according to the preferred species and equalization circuitry.

Figure 12 is a block diagram of the preferred species of a receiver within the inventive genus which can receive data transmitted by the transmitter of Figure 11 and supports TDMA/CDMA spreading, code deshuffling supporting code diversity, forward error correction, equalization, and Viterbi Decoding.

Figure 13 is a block diagram of the preferred form of frame detector 882 in the transmitter of Figure 12 for achieving receive frame synchronization and chip clock synchronization.

Figure 14 is a timing diagram showing how the frame/ranging detector 880 does coarse tuning to find the gap in the frames transmitted by the CU.

Figure 15 is a diagram illustrating how the early late gating sampling is used to find chip clock synchronization (showing the situation when chip clock synchronization has been achieved.

Figure 16 illustrates the 3 permissible patterns of data at the output of comparator 950 for a centered barker code condition to be declared.

Figure 17 is a block diagram of the preferred form of modulator using raised cosine shaping filters whose transfer functions are Hilbert transforms of each other.

Figure 18 is a block diagram of a system according to a broad teaching of the genus of the invention.

Figure 19 is a block diagram of a synchronous TDMA system for bidirectionally communicating digital data over any transmission media including hybrid fiber coax using FDMA upstream and downstream channel separation so as to not interfere with other services such as cable television programming sharing the HFC.

Figure 20 is a block diagram of the basic functional blocks needed at the CU and one RU to carry out ATM protocol communication using virtual links on a CATV HFC cable plant.

Figure 21 is a symbolic diagram of a typical CATV HFC.

Figure 22 is a diagram of how virtual links look at the logical level despite the physical media structure shown in Figure 21.

Figure 23 is a diagram illustrating the downstream data structure.

Figure 24 is a diagram illustrating a Utopia + format 55 byte ATM cell as is used in the downstream data and illustrating how a 9th bit is added for use in signalling the beginning of each ATM cell and to carry CRC data.

Figure 25 is a diagram illustrating how the bytes from one upstream ATM cell from one RU are interleaved into the frames with bytes from other RUs in accordance with a hypothetical code allocation of codes 1, 5 and 35 for one RU.

Figure 26 is a diagram illustrating the various layers of the OSI model and illustrating what jobs/protocols are done on each layer by the system of the invention.

Figure 27 is a diagram illustrating the interfaces between the upper OSI model layers at the CU and RUs with the data link, media access control and physical layers implemented by the system of the invention including connection between these layers by the virtual network.

Figure 28 is an illustration of the format of a Utopia + 55 byte ATM cell including a two byte virtual link header enabling use of this cell in a CATV HFC plant.

Figure 29 illustrates the SCDMA frame structure of a superframe including 4 frames separated by guardbands, each frame being 125 microseconds long and containing data from 144 timeslots.

Figure 30 illustrates how inband management channels are fit into every frame with 128 payload data timeslots.

Figure 31 illustrates the physical layer downstream broadcast architecture for contention resolution and channel allocation messages.

Figure 32 illustrates the upstream multipoint to point multiple access communication architecture.

Figure 33 illustrates the format of a typical upstream management and control message.

Figure 34 illustrates access request collisions and contention resolution protocols.

Figure 35 illustrates the physical layer initialization sequence.

Figure 36 illustrates one embodiment of a dynamic bandwidth manager process carried out by the CU computer.

Figure 37 illustrates how bandwidth allocation is dynamically altered in accordance with bandwidth actually used.

Figures 38A, 38B and 38C comprise a flow chart illustrating more details of one embodiment for the dynamic bandwidth reallocation process symbolized by block 1108 in Figure 36.

Figure 39 illustrates one possible structure for a table of data for use by the process of Figures 38A-38B in dynamically reallocating bandwidth.

Figure 40 is a block diagram of the preferred embodiment for the multiplexer/demultiplexer ATM/SCDMA interface 1009 in Figure 20.

Figure 41 is a flowchart of operations by the data framer 1246.

Figure 42 is a flowchart of the processing carried out by the cell output controller 1243 in retrieving completed ATM cells from the cell buffer memory 1254 and transferring them to the SAR.

Figure 43 is a block diagram of the circuitry inside a formatter 1030 inside every RU modem.

Figure 44 is a block diagram of an optimized system according to the teachings of the invention to send data organized into ATM format cells over a CATV system bidirectionally between a CU modem and a plurality of RU modems using SCDMA with the ATM cells optimized to have 2 byte headers in the downstream direction and 0 byte headers in the upstream direction.

Figure 45 is a block diagram of an RU modem having a structure designed for use with a number of peripherals coupled to the RU modem by an Ethernet LAN.

Figure 46 is a diagram of an optimized downstream ATM cell having a 2-byte header.

Figure 47 is a diagram of an optimized upstream ATM cell having a 0 byte header.

Figure 48 is a more detailed block diagram of the circuits inside blocks 1370 and 1372 in Figure 44.

Figures 49A through 49C are a flow chart illustrating the detailed steps of the downstream data transmission process using optimized ATM cells.

Figures 50A through 50I illustrate packet and cell formats at various stages of the optimized downstream processing.

Figure 51 shows the software architecture within each node coupled to an RU modem including the IP layer and the Ethernet layer.

Figures 52A and 52B are a flowchart illustrating the steps in the upstream transmission of data from multiple sources to the CU using an ATM protocol and SCDMA.

Figures 53A and 53B are a flow chart illustrating the process of using the DHCP protocol by each RU modem at powerup time to obtain an IP address and how this IP address is bound to the Ethernet address for each RU in the router tables at the CU.

Figure 54 is a block diagram for a CU modem for an alternative embodiment wherein any other form of multiplexing is used in the upstream direction to establish the virtual links needed to support ATM cell transmission over shared media 1000.

Figure 55 is a block diagram of an RU modem for the alternative embodiment in which the CU modem of Figure 54 is used.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

Code Division Multiple Access System For CATV Media

Many of the individual concepts utilized in systems according to the teachings of the invention are known in the prior art and are described in detail in Dixon, *Spread Spectrum Systems with Commercial Applications* (3rd Ed. 1994) Wiley & Sons, ISBN0-471-59342-7, and Haykin, *Communication Systems* (3rd Ed. 1994) Wiley & Sons, ISBN 0-471-57178-8, both of which are hereby incorporated by reference.

Use of Cyclic Codes in Code Division Multiple Access For Better Performance

In the preferred embodiment, the orthogonal codes used in the modulator/transmitters are cyclic codes. In cyclic orthogonal codes, all codes used are the same sequence of numbers, but each code is shifted by one or more bit positions from the preceding code. Although any set of orthogonal codes will work to implement the invention, the cyclic orthogonal codes simplify implementation issues by reducing the amount of storage needed to store the codes.

Those skilled in the art will appreciate that each subscriber transmitter may transmit multiple channels of digital data, and that the matrix multiplication and summation operations described above may be performed with digital circuitry such as suitably programmed microprocessors.

In an alternative embodiment, the separate streams of digital data are transmitted using spread spectrum frequency hopping techniques. In this embodiment, a first stream of digital data will be transmitted from one end to the other using a carrier that hops in frequency in accordance with a first predetermined coded sequence.

Synchronous CDMA: The Alignment/Ranging Process to Achieve Frame Synchronization

Alignment is an important issue for optimal operation of the system with minimal cross talk between channels. In the system, the time slots in TDMA streams are the channels. The digital data in the TDMA streams is re-arranged into symbols, and is transmitted in frames, with three symbols plus one guard band or gap per frame. The guardband or gap is reserved for transmission of alignment barker codes, and no other data is supposed to be transmitted during the gaps.

The concept in alignment is to adjust variable delays imposed at the site of each transmitter prior to transmission of a barker code so as to compensate for different

propagation delays from each transmitter site such that the barker code from each subscriber transmitter trying to align arrives at the CU receiver during the same gap. When the variable delays at each subscriber transmitter are adjusted properly, each subscriber will be said to be in alignment so that the signals encoding the symbols that are simultaneously transmitted on the shared data path will all be transmitted with the same frame timing.

Alignment is important to obtain pure orthogonality so as to obtain zero cross talk. If the transmitters are not perfectly aligned, the signals transmitted can still be recovered, but there is some cross talk between channels which will limit the capacity of the system to carry information.

This process of aligning all the delay circuits in the transmitters is sometimes alternatively called ranging herein and is broadly applicable to other types of multiple access digital data transmission systems also which suffer from different propagation times from different transmitter sites such as time division multiple access systems that form part of the prior art discussed above.

Referring to Fig. 1A, there is shown a diagram of the typical frame structure. In the preferred embodiment, each frame is composed of three symbols of 144 chips each and a gap or guardband comprised of 16 chips for a total of 448 chips each having 278 nanoseconds duration. The chip is the basic unit of time in the "code domain", where code domain refers to the signals propagating across the shared media. In the preferred embodiment, each chip is a QAM modulated element of a result vector where the result vector is comprised of a number of elements equal to the number of timeslots and is the result of code division spreading of the elements of an information vector constructed from the bits of each channel or timeslot. In the preferred embodiment, each receiver receives a TDMA serial bit stream comprised of 144 individual timeslots or channels each of which contains 8 bits. To these 8 bits there is added a 9th bit in the preferred embodiment which can be used for side channel conversations with the CU unrelated to the data received from the external device. These 9 bits are divided into three tribits of 3 bits apiece. A collection of 144 of these tribits is stored in a framer memory and, these 144 tribits will be the information vector which is multiplied by the code matrix to generate a result vector having 144 elements. These 144 result vector elements will be QAM modulated to generate the 144 chips that are transmitted as a symbol. This process is repeated for each of the three tribits of each timeslot thereby resulting in the transmission of three symbols in each frame. In the preferred embodiment however, each tribit is encoded with one or more redundant bits based upon the three bits and the state of these same three bits of the same timeslot during the last frame. The redundant

bit(s) is calculated to aid a Viterbi Decoder in a receiver in the central unit to ascertain with a higher degree of accuracy from the received signals which have been corrupted by media impairments what bits were originally present as each tribit.

One skilled in the art will appreciate that the construction of the information vector which will be used to generate each symbol by taking only some of the bits from each timeslot spreads the data from each timeslot out over time. This renders the data less susceptible to burst noise. The code division multiplexing allows multiple channels of digital data to be simultaneously transmitted in a 6 MHz channel without interference between channels. In addition, frequency division multiplexing may be utilized to transmit even more channels of digital data above and beyond the 144 channels transmitted in the first 6 MHz channel. In other words, another 144 different TDMA digital channels may be code division multiplexed and transmitted simultaneously with the first 144 digital channels but on a second 6 MHz channel. This second 6 MHz channel has a different center frequency than the first 6 MHz channel which is separated from the center frequency of the first 6 MHz channel sufficiently to not interfere therewith. Both the first and second 6 MHz channels have center frequencies which are separated sufficiently from the center frequencies of the cable television programming sharing the same media so as to not interfere therewith.

In Fig. 1A, the three symbols of frame F_n are symbolized by blocks 62, 64, and 66. The gap or guardband is symbolized by blocks 60 and 71. There is one guardband associated with each frame. The guardband 71 (sometimes also referred to herein as the gap) is used for synchronization and equalization purposes for the frame comprised of symbols 62, 64, 66 and guardband 71. The symbols carry the information for the various channels of digital data provided to the subscribers. The frame period is 125 microseconds. The frame data payload is 128 channels times 72 kilobits per second per channel plus 16 control and management channels each of which has a data rate of 72 kilobits per second for management and control information.

Hereafter, each subscriber transmitter will be referred to as a remote unit or RU, and the central unit or CU will be referred to as the CU.

The process of synchronization is the process wherein each RU is "trained", i.e., has a variable delay in its transmitter set using feedback from the CU on management and control channels such that the transmitted frame from each RU arrives at the CU at the same time. Alignment of all frames from all RUs results in the beginning of the gap 60 for each frame from each RU occurring at the same time at the location of the CU regardless of differences in propagation delays from the various RUs to the CU. In

Figure 1A, time increases to the right. Therefore the beginning of the guardband 60 is located at point 61.

Alignment of Any Digital Data System That Sends Data Bits Collected As Frames

Referring to Figure 1B, there is shown a symbolic diagram illustrating the concepts involved in alignment. In Fig. 1B points having increasing positive coordinates along the y-axis starting from the origin at 100 represent increasing time. Points along the x-axis to the right of origin represent increasing distance from the central unit which is designated at position 70. Time 100 represents the beginning of symbol 62 in Figure 1A at the CU. The gap 71 at the end of the three symbols will be used for alignment, and the end of gap 71 will be deemed the end of the frame.

The alignment process is started asynchronously by any RU that needs to align. The central unit transmits a barker code during each frame at the same time in the frame. This barker code is received by each remote unit at a different time because of different propagation delays, but as to any particular RU, the barker code is always received at the same time during every frame until the CU changes its delay (a concept to be discussed more fully below). The barker code represents a trigger to any RU attempting to align and marks the receive frame timing reference for that RU. The time of receipt of the barker code represents the start of the variable delay interval being adjusted by the RU during the alignment process.

The CU's "every frame" barker code transmission during the frame shown in Figure 1B is represented by line 80. The barker code is received by RU #1 at position 67 at time 72. The barker code is received by RU #2 at position 69 at time 74. The alignment process is a trial and error process of adjusting a delay from the time of receipt of the barker code to the time of transmission of the same barker code by each RU back toward the central unit 70 until the delay is properly adjusted such that the re-transmitted barker code arrives at the CU during the gap. Vector 68 represents correct delay timing for RU #1 at position 67 such that its barker code transmission 73 arrives in the middle of the gap 71. Dashed vector 76 represents an incorrect delay resulting in a barker code transmission, represented by dashed line 78, from RU #1 which arrives sometime during the middle of symbol 66 thereby missing the gap 71. This condition represents an incorrect alignment and may result in crosstalk.

Likewise, the RU #2 at position 69 uses zero delay and emits a barker code transmission 82 immediately upon receipt of the barker code trigger transmission 80 from the CU 70. This barker code transmission 82 from RU # 2 also arrives during the

middle of gap 71 thereby indicating that RU # 1 and RU # 2 are correctly aligned.

The alignment barker code transmissions are typically short bursts having energy levels which are sufficient to make detection during gap 71 easy even though gap 71 also includes random noise energy.

The alignment barker code transmissions are detected during the gap by performing a correlation mathematical operation in the CU receiver between the barker code that was transmitted and the received signal. If the received signal was the same barker code that was transmitted by the CU, the correlation operation will output a signal that peaks at the time of maximum overlap between the barker code transmitted by the CU and the received signal. The timing of this peak indicates the alignment state of the RU that transmitted the barker code which resulted in the peak. Because the barker code transmissions are relatively short in duration and their amplitudes are not excessive, arrival of a barker code transmission during the middle of a symbol will generally not cause errors in the interpretation of symbol 66 by the CU receiver. Each symbol encoded in the code domain includes error detection and correction bits (ECC bits) such that any errors that occur can usually be detected and corrected when the symbols are re-constituted by the framer circuitry in the receiver. Therefore, if the barker code alignment transmission does result in an error, that error will usually be within the detection and correction range of the ECC bits of each symbol.

Ranging Process

Referring to Figure 2, which is comprised of Figures 2A, 2B, and 2C, there is shown a flow chart for the general alignment/ranging process which is used in training all RUs to set their transmit frame timing delays T_d properly such that each frame transmitted by an RU will arrive at the CU at the same time as all other frames transmitted from other RUs despite differing propagation times. One of the unique characteristics of the ranging processes described herein is that the RU does the ranging process and the CU is more or less passive which is in contrast with the prior art.

Generally at the time of powerup of an RU, the RU first adjusts its AGC level to make full use of its analog to digital converter dynamic range. Next, the RU does frame detection to determine when the gaps in the CU broadcasts are by performing correlations in the RU receiver frame detector looking for the known barker code which the CU transmits during every gap. Once the gap is located, the frame detector sets the time base generator to synchronize on that receive frame timing reference. Next, the RU performs chip clock synchronization and carrier recovery in the manner described below in the discussion of Figure 3. Carrier recovery is done by examining slicer error

on a known BPSK pilot carrier or pilot channel signal transmitted during a predetermined timeslot using a predetermined code (CU local oscillator signal samples in timeslot 0 spread with all 1s CDMA code and transmitted using BPSK). The pilot channel also carries frame number data. The slicer error is used to synchronize the RU local oscillator to the phase of the CU local oscillator. Chip clock synchronization is performed by the fine tuning circuitry of the frame detector from the chip clock embedded in the barker code sent by the CU during every gap. This is all the RU needs to do to set itself up for reception of CU data and messages.

The RU then starts listening to CU messages to determine if it tuned to the right CU and to determine when the CU solicits ranging activity by a message on one of the command and control channel. In some embodiments, the "clear to range" message can be eliminated, and the CU can watch for ranging barker codes all the time, but it is preferred to allow the CU to throttle ranging activity. The RU then performs a ranging process described below and registers itself with the CU by sending an authentication sequence of barker codes after frame synchronization has been achieved (discussed below). This is done by the CPU when it receives a message via C3 circuit 860 from the CU saying "I found one barker code in the gap, please send your authentication code". The CPU then sends data on bus 512 to ranging circuit 510 in Figure 11 telling it what authentication barker code sequence to send. The CU will then transmit a message indicating what authentication code it found and how many chips off center of the gap the barker code is. The CPU 405 in the RU that is ranging then properly adjusts the transmit frame timing delay reference T_d on bus 499 to center the barker code in the gap. Other items of data the CPU 405 sends to the ranging circuit 510 is data labelled P indicating the power level to use for the ranging barker codes and an RU/CU signal indicating to the ranging circuit 510 whether it should follow the rules of ranging for an RU or CU.

The CU next instructs the RU to entering an equalization training interval to determine the coefficients to set into the RU transmitter's precode filter such as filter 563 in Figure 11 to predistort the RU signals to eliminate channel distortion and test the quality of the ranging result.

The ranging process starts as symbolized at block 180 with the CU waiting for a predetermined interval from the start of each frame and then sending a trigger signal barker code transmission to the RUs during the gap. Usually this trigger signal is sent during the gaps between frames even when the CU adds additional delay for reasons discussed below. The RUs monitor these gaps for these barker codes using their frame

detector circuits such as circuit 513 in Figure 3 and circuit 882 in Figure 12.

Block 182 symbolizes the process wherein each RU trying to synchronize (the terms "synchronize", "ranging" and " alignment" all are used synonymously to mean the process of training an RU to set its delay vector properly to get its frame boundaries aligned with the CU frame boundaries) receives the barker code trigger signal transmission from the CU and sets its receive frame timing and then sets a first delay for its delay vector. Thereafter, the RU transmits, during the RU's interframe gap, the same barker code it received from the CU towards the CU as an alignment transmission.

In block 184, the CU monitors the gap for activity by performing a correlation mathematical function between any received signal during the gap and the barker code that was transmitted as the trigger signal. If a barker code identical to the trigger signal is received during the gap, the correlation calculation will result in a correlation peak being found in the gap. If the correlation calculation results in a peak being found, processing proceeds to the process symbolized by block 190. There, the CU broadcasts a message to all RUs indicating that it found activity in the middle 8 chips of the gap (or anywhere in the gap in some embodiments). Then the process of block 192 is performed where each RU trying to synchronize sends its "signature", i.e., its RU identification code in the form of a barker code transmission sequence. That is, in response to the broadcasts from the CU, each RU trying to synchronize sends its unique signature towards the CU in order to determine if that RU's barker code is the barker code the CU found in the gap and whether it is the only RU in the gap. This process is called authentication.

The process of block 192 symbolizes the start of the authentication process. Each RU has a unique signature which comprises the transmission and nontransmission of barker codes during the gaps of a multiple frame authentication period. Specifically, the unique signature of each RU will involve transmitting the barker code during some gaps of the authentication period but not during others in a sort of "Morse code". Each barker code transmission results in a correlation peak during one of the 8 chips in the middle of the gap. Each RU has a unique 16 bit RU ID, each bit being either the presence or absence of a barker code correlation peak somewhere in the middle 8 chips of the gap (or anywhere in the gap in some embodiments). Therefore, it takes 16 frames or 4 suprerframes to transmit the RU ID. The number of gaps during which the barker code is transmitted compared to the number of gaps during which the barker code is not transmitted during the authentication period is such that if only one RU is aligned to the gap and is transmitting its authentication signature, activity will be found in the gaps of the authentication interval only 50% of the time. This scheme for authentication is

chosen so that the CU can detect contentions, i.e., more than one RU in the same gap, in the manner described below.

After performing the process of block 192, the process of block 194 on Figure 2B is performed. This process involves the CU monitoring each of the gaps during the plurality of signature sequence frames in the authentication interval and performing correlations between the signals received in each of the gaps and the barker code that the CU transmitted. Correlation peaks are found comparing the correlator output to a threshold value. The threshold value is set by detecting a noise threshold when the gap is empty and setting the threshold at a fixed delta above the empty gap base noise value.

Next, the process of block 196 is performed. In this process, the CU counts the number of gaps in the authentication interval that have had activity detected therein, and then compares that number to the total number frames in the authentication interval to determine if the 50% activity level limit has been exceeded indicating that more than one RU is hitting the gap. The advantage of this method is that activity detection, contention detection and authentication are all combined into a single process thereby speeding up the process by more efficiency.

Returning to the consideration of the process of block 184, if the CU, while monitoring the alignment gap for activity, finds no peak resulted from the correlation calculation, then the process of block 186 is performed. In the process of block 186, the CU broadcasts a message to all RUs telling them to adjust their delays and to try again to hit the gap with their barker code transmissions. Then, the process of block 188 is performed wherein each RU trying to synchronize increments its delay vector and retransmits the same barker code as was received from the CU. Thereafter, the process of block 184 is performed again wherein the CU monitors the gap for activity. The loop comprising blocks 184, 186 and 188, taken together, comprise the trial and error process which causes all RUs trying to align themselves to continually increment their delay vectors until at least one of them hits the gap.

Returning to the consideration of block 196, if 50% activity level is detected during the authentication interval, it means that only one RU is in the gap. In such a case, the process of block 198 is performed. In this process, the CU identifies the RU whose barker code transmissions are found in the gap from the unique signature sequence transmitted during the authentication interval. In other words, the CU examines exactly which gaps had correlation peaks therein and the sequence of these gaps and looks up this sequence in a lookup table listing the unique signature sequence for each RU in order to identify the particular RU that has successfully aligned itself. Block 198 is reached only if activity is detected in exactly 50% of the gaps.

After the CU identifies the RU, it broadcasts the identity so determined to all RUs as the last step of block 198.

Next, the process of block 200 is performed. In this process, the RU with the identity broadcast by the CU recognizes its identity in the broadcast message and enters a fine tuning mode.

The fine tuning mode is represented by the process of block 202. In this process, the CU instructs the RU which has aligned itself in the gap on how to adjust its delay vector in order to center the correlation peak calculated by the CU to the exact middle of the gap. In the preferred embodiment, the gap is comprised of 16 chips which comprise 8 chips in the middle of the gap and then 4 chips on either side of this middle group of 8. It is desirable during the fine tuning mode to get the correlation peak centered in the middle of the middle 8 chips. As mentioned above, a chip is a small interval of time equal to the frame period of 125 microseconds divided by the 448 chips which comprise each frame. In other words, each chip is 279 nanoseconds in duration. The fine tuning process of block 202 involves sending messages back and forth between the CU and the RU which has been identified as having aligned itself in the gap. These messages are sent over the management and control channels. Usually the exchange involves only one instruction from the CU to the RU saying, for example, "Increase your delay vector by 2 chips" or , "Decrease your delay vector by 3 chips". The RU then makes the instructed adjustment and retransmits the barker code. The CU again calculates a correlation peak and examines where the peak occurs in the gap. If the peak occurs in a suitable position, the CU sends a message to the RU telling it to stop adjusting its delay vector as satisfactory alignment has been achieved.

Returning to the consideration of the process of block 196, if the CU determines that greater than 50% of the gaps during the authentication interval had correlation peaks therein, i.e., greater than 50% activity is detected, then the process of block 204 is reached. This process is only reached if more than one RU has aligned itself to the same gap. If this case, because each RU is transmitting its unique signature, and because each signature is a unique sequence with only 50% activity level, the result of two RU's being in the same gap will be that during more than 50% of the gaps of the authentication interval, correlation peaks will occur. It is impossible to find tune the RUs if more than one RU is trying to fine tune during the same gap. Therefore, the CU has to reduce the number of RUs that are in the gap to one, and it starts this process by performing the process of block 204. In this process, the CU broadcasts a message to all RUs instructing only the RUs attempting to synchronize to execute their collision resolution protocols.

Next, the process of block 206 is performed, to start the collision resolution protocol, wherein each RU attempting to synchronize executes a random decision whether to continue attempting to synchronize or to stop attempting to synchronize. Each RU will make this decision with a 50% probability of either outcome.

After all RUs make their random decisions whether to continue, the process of block 208 is performed. In this process, the RUs that have decided to continue to align retransmit their signature sequences without changing their timing, i.e., with the same timing as was used on the last iteration of the trial and error process. In other words, each RU that has decided to continue transmits its unique signature sequence (sometimes hereafter called a "dotted sequence") over another authentication interval using the same delay vectors that are currently set.

Next, the process of block 210 on Figure 2C is performed wherein the CU again monitors the gaps of the authentication interval for activity.

If the random decisions whether to continue or not result in no RUs transmitting their signatures, then no activity will be found in the gaps of the authentication interval. In this event, the process of block 212 will be performed wherein the CU broadcasts a message instructing all RUs to go back to the previous stage and to reexecute their decisions to continue or discontinue the ranging process.

The RUs then re-execute their decisions whether to continue or stop attempting to align themselves and retransmit their signatures during the authentication interval with the same delay timing used on the previous iteration, as symbolized by block 214.

Following the process of block 214, the process of block 216 is performed to determine if more than 10 attempts to get one RU in the gap have occurred. If so, the process of block 218 is performed to return to block 180 and restart the ranging process from the top. If fewer than 10 attempts have been made, processing returns to the process of block 210 wherein the CU again monitors the gaps of the authentication interval for activity.

If the process of block 210 finds only one RU in the gap, i.e., 50% activity level is detected during the authentication interval, then the process of block 222 is performed. The process of block 222 authenticates the RU by broadcasting the identity of the RU found in the gap and then the RU is fine tuned in the manner previously described with reference to block 202.

If the CU finds in the process of block 210 more than one RU is still in the gap, processing returns to block 204 where the CU broadcasts a message to all RUs instructing them to execute their collision resolutions protocols. This process is symbolized by block 220.

Note that in the ranging process described above, it is the RUs that determine how far they are from the CU rather than the CU determining how far each RU is from it. The advantage of having the RUs doing the ranging is that the CU does not have to stop traffic on the various channels to perform ranging functions each time a new RU enters the system or an existing RU loses synchronization. In a system where the traffic may frequently include high demand applications such as real time video, stopping traffic flow for ranging is not a viable possibility because it would interrupt the flow of video information and disrupt the subscriber's video conference, movie etc. In the ranging system described herein in its various embodiments, there is no need to stop traffic since the ranging process is done out of band, i.e., in the gaps. Further, because the transmitted power of the barker codes is low and correlation processes are used, the process can start blind with any trial and error timing value without interfering with channel traffic. That is, even if the barker code transmitted back toward the CU by the RU has improper timing and lands somewhere outside the gap, its power level is low enough to not cause substantial interference, and even if some small amount of interference is caused, the chips of the symbols transmitted during the frame have enough redundancy with the trellis encoded modulation to recover from the interference without an error. Because correlation to a known barker code pattern (the same barker code pattern the CU transmitted to the RUs during the previous gap) is used by the CU to determine whether it has or has not detected a barker code from an RU in the gap, the RUs can transmit their barker codes at very low power levels so as to avoid interfering with traffic and causing errors in the data of the various payload channels during the trial and error process of setting their transmit frame timing delay values T_d so as to hit the gap.

In other embodiments however, conventional ranging techniques could be used where the CU measures the range to the RUs to establish synchronous CDMA.

In the high power pulse embodiments described above, the RUs act like transponders by sending a narrow, high amplitude pulse upon receipt of a trigger signal from the CU. The trigger signal from the CU could be a special pulse, a barker code, etc. If the RU was misaligned, and the large amplitude pulse landed in the middle of the upstream payload data, the CU would ignore the particular chip which was "stepped on" by the high amplitude pulse. The payload data could still be recovered because the bandwidth of the payload data has been spread so widely using direct sequence CDMA spreading. Trellis code modulation is not needed for this scheme to work. After detecting the RU's pulse and comparing its timing with the position of the frame timing reference,

the CU would ask the RU for its identity and the RU would send it by any conventional manner such as pulse position modulation, amplitude shift keying etc. The CU would then send a message to the RU instructing it to change its transmit frame timing delay in a direction to put the pulse closer to the fixed timing reference, and this process would continue until the RU hit the timing reference. Note in this method, that a gap or guardband is not needed in each frame.

Channel allocation by the CU can take any one of a number of different forms. For example, the RUs could have a fixed allocation of channels or channels could be awarded in any number to any RU based upon need where the CU polls the individual RUs for their needs or the RUs transmit their needs asynchronously to the CU and the CU arbitrates between the requests to allocate the available channels. Likewise, one RU may have security considerations the require one channel to be dedicated to it at all times and no other RU is allowed to be on that channel as controlled by channel awards by the CU given in messages to the individual RU's. Alternatively, some channels can be made available for all RUs to use with the RUs themselves resolving contentions. In the preferred embodiment, there are four channel allocation schemes which are implemented either individually or in any combination in the CU channel allocation circuitry: (1) a reservation scheme where the RUs bid for bandwidth and the CU reserves certain channels to each of the RU's; (2) a contention mechanism where the RUs are notified by the CU of what channels are available to all RUs for traffic, and where the RUs transmit on those channels at will with contentions detected by the CU and contention notification messages to the RUs in contention to enter contention resolution procedures; (3) polling where the CU inquires of each RU sequentially whether it needs bandwidth and awards bandwidth as needed as determined from the polling with arbitration when not enough channels are available to meet all requests; (4) fixed allocation of the available channels to specific RU's. In the preferred embodiment, all four schemes can be used individually at times or any combination of the schemes can be used at times. Which channel allocation schemes are in use at particular times is established by the configuration data set up by the user. Synchronous TDMA multiplexing schemes are described in "*Data and Computer Communications*" by Dr. William Stallings, at page 211-213, Macmillan Publishing Co., New York (4th Ed. 1994) ISBN0-02-415441-5 which is incorporated by reference herein.

A Code Division Multiple Access Transceiver

The genus of the invention contemplates a synchronous code division multiple access system for use on a CATV system to provide supplemental digital services wherein all the bandwidth dedicated to the supplemental services is continuously completely used

and may be shared simultaneously by multiple users. Specifically, a plurality of orthogonally encoded, pseudo point-to-point channels are provided which may be shared by all users. The data on each channel is sent in frames. The users and RU transceivers are physically distributed along the CATV system thereby causing differing propagation times to the CU for each user both by virtue of physically different paths to the CU as well as by network thermal expansion and contraction. A training interval is used wherein each RU performs a trial and error process to set its transmit frame timing delay value T_d to a delay which results in frames from that RU arriving simultaneously at the CU with frames transmitted by all the other RUs. This frame synchronization maximizes the number of users which can share the available bandwidth by reducing crosstalk between codes. In species within this genus, trellis encoded modulation and Viterbi decoding is used to lower the error rate in the face of the channel impairments. Quadrature amplitude modulation is also used in the trellis encoded modulation species and other species within the genus of the invention to maximize bandwidth efficiency. In other species within the genus, a training period is used to learn channel impairments then existing for each RU, and these channel impairments are converted to coefficients that are fed to precode filters so as to set the transfer functions thereof so as to predistort the outgoing signals so that they arrive relatively free of distortions caused by channel impairments.

Referring to Figure 3, there is shown a high level block diagram of the preferred species of a transceiver for use in the modem of each RU and CU. The transmit channel of the transceiver uses a framer circuit 400. The function of the framer is to receive one or more streams of digital data via data path 399 from one or more sources and to organize this data into a plurality of frames, each frame comprised of one or more symbols. In the preferred embodiment, the framer circuit 400 composes the frames of data from a TDMA data stream on bus 399 where each timeslot corresponds to one channel. There are 128 payload data channels to share and 16 management and control channels some of which are also shared for a total of 144 channels or timeslots. Each RU may be assigned one or more channels or timeslots depending upon the amount of bandwidth it has been awarded by the CU in response to requests for bandwidth from the RU. In addition, bandwidth may be reserved to the various RUs on a permanent basis in some embodiments, and in these embodiments, the channels or timeslots may be permanently assigned or the reserved number of channels may be assigned on a guaranteed basis each time the RU requests bandwidth.

It is not critical to the invention that the incoming data streams arrive in a TDMA stream on bus 399. The streams of data from peripheral devices or networks could, in alternative embodiments, arrive via FDMA on bus 399 or each source of data could be connected to the framer circuit 400 by a separate input bus.

The framer circuit 400 and its associated circuitry implement the variable delay that sets the transmit frame timing reference for each RU and CU. This transmit frame timing reference establishes the timing of transmission of the orthogonally CDMA encoded chips of each frame such that all frames arrive from the each of the physically distributed RUs at the CU at the same time. Although, the invention still works even if frame synchronization is not maintained because of the orthogonality of the CDMA codes which are used, it does not work as well since the maximum number of users which can be simultaneously be sharing the available payload channels is limited. This is because there are higher levels of crosstalk between CDMA codes when frame timing synchronization between all RUs and the CU is not maintained. Therefore, each RU undergoes a training interval after first powerup and from time to time thereafter to set its transmit frame timing delay. The CPU 405 changes the value of T_d on line 499 until frame synchronization is achieved and thereafter maintains whatever value of T_d which resulted in frame synchronization having been achieved.

The particular manner in which frame synchronization is achieved is not critical to the invention.

The framer circuit 400, in the preferred TDMA input bus embodiment, bridges the two time domains between the TDMA input data and the chip clock code domain (reading of the framer circuit is done at the chip clock rate and writing is done at the byte clock rate at which timeslots of data are written one 9-bit byte at a time). The output data stream from the framer circuit 400 comprises three arrays of tribits per frame, each array of tribits representing an information vector which, after encoding by the orthogonal multiplexer 408, is transformed into one symbol of chips. In the preferred embodiment, the orthogonal multiplexer 408 is a code division multiplexer which uses a plurality of orthogonal codes, each code being used to encode the data from a different channel. This is a so-called direct sequence type spread spectrum operation wherein the bandwidth of the baseband signals on buses 1068C and 1070C are spread across a broad spectrum by the CDMA codes using orthogonal code multiplexer 527 in Figure 11 and orthogonal code multiplexer 408 in Figure 3.

In an important class of alternative embodiments, the orthogonal encoding multiplexer 408 (and orthogonal multiplexer 527 in Figure 11) could be any encoder

which encodes each channel with a different orthogonal waveform. For example, these orthogonal multiplexer could store digital samples that define a plurality of orthogonal sine and cosine waveforms, each at a different frequency. Any other set of orthogonal waveforms of different frequencies other than sines and cosines would also work to encode the various channel data samples. Each channel's data would then be multiplied by a different waveform's samples to generate new digital samples which define orthogonally encoded data on buses 417 and 419 for modulation onto the RF carrier frequencies. In such embodiments, the bandwith of each channel's data is not spread as wide as in a CDMA system. In fact, each channel's data would be dumped into a narrow bandwidth frequency bin. In such systems, the orthogonal demultiplexers 462 in Figure 3 and 766 in Figure 12 would perform the inverse transformation on the received samples to bring them back to baseband signals on bus 463 in Figure 3 and bus 776 in Figure 12. For example, each of orthogonal code multiplexer 527 in Figure 11 and orthogonal code multiplexer 408 in Figure 3 could be an inverse Fourier transform processor. The inputs to the inverse Fourier transform processors 408 in Figure 3 and 527 in Figure 11 in this alternative embodiment would be the information vector elements on buses 1068C and 1070C in Figure 3 and buses 549R and 549I in Figure 11. Each of these information vector elements would define the magnitude of one frequency component in the Fourier spectrum of the output signal to be generated. The inverse Fourier transform processor would then calculate the time domain waveform that would have that Fourier spectrum and output digital samples that define that time domain waveform on buses 558R and 558I in Figure 11 and buses 417 and 419 in Figure 3. These samples would be used to modulate one or more RF carriers in accordance with whatever modulation scheme was being used. The receiver's demultiplexers (462 in Figure 3 and 766 in Figure 12) then perform a Fourier transform on the incoming signal samples to output the individual frequency component magnitudes that define the original information vector components.

Note that each information vector element in this embodiment always defines the magnitude of the same frequency component. In an alternative multitone system, the information vector elements can be pseudorandomly scrambled in the transmitters so that they define different frequency component magnitudes in each frame and then pseudorandomly descrambled in the same order in the receivers.

In SCDMA direct sequence spread spectrum transmitters of the preferred embodiment, the three information vectors output during each frame are converted by CDMA spreading to the three symbols that are transmitted during that frame. The data in each information vector spans the entire 144 timeslots in the sense that three bits from

each timeslot or channel are present as the elements of the information vector as a tribit. This interleaving of data from each timeslot into each information vector is preferred but not critical to the invention. Likewise, the transmission of three symbols per frame is not critical to the invention and fewer or greater numbers of symbols could be transmitted.

In the preferred embodiment, the circuitry of the transceiver is virtually all digital, so the arrays of tribits are true arrays, the elements of which are used sequentially in the matrix multiplication to perform the CDMA spreading.

In analog embodiments, the arrays of tribits will be streams of tribits, with three separate streams per frame.

The relationships between timing in the time domain and timing in the code domain are as follows:

- There are 144 total time slots or channels in the TDMA stream, of which 128 are payload time slots and 16 are management and control time slots;
- Each time slot or channel in the TDMA streams carries 9 bits of digital data synchronized with the bit clock;
- One time slot worth of data or 9 bits is stored in the framer for each cycle of the byte clock;
- 1 frame = 144 times slots, each with 9 bits plus 16 chips for the alignment gap;
- 1 frame also equals 3 symbols plus the 16 chip periods of the alignment gap = 448 chip periods;
- 1 symbol = 144 chip periods;
- 1 gap = 16 chip periods;
- For every 16 bit clock periods, there are 7 chip clock periods, and for every byte clock period, there are 9 bit clock periods.

Figure 4 is a diagram which helps illustrate the manner in which framer memory 300 is emptied for transmission. Figure 4 shows a completely filled page comprising 144 memory addresses, each filled with one 9-bit byte, and divided into three columns of 3-bit tribits. Each column, marked by the legends symbol 1, symbol 2 and symbol 3, is comprised of 144 tribits and represents one symbol of a frame. To send this frame of data, the read pointer will increment 144 times during the time the first symbol is being encoded. The state of the tribit select counter during this first 144 cycles is such that only the 144 tribits of symbol 1 will be output on bus 360 to the forward error correction (FEC) encoder 402 in Figure 3.

After the 144th incrementation, the read pointer counter 324 rolls over to zero and begins to count up to 143 again. At the 144th incrementation, the tribit select

counter increments which causes the multiplexer 356 to select the middle column of tribits from symbol 2 in Figure 4 for output to the forward error correction encoder 402 in Figure 3. A similar process unloads the 144 tribits of symbol 3.

Returning to the consideration of the transceiver block diagram of Figure 3, the output data streams from the framer 400 on bus 360 may optionally be passed through a forward error correction encoder 402. The forward error correction encoder 402 can be eliminated in some embodiments or an ARQ encoder may be substituted. The embodiment of Figure 3 symbolizes a class of species which use systematic codes where the bits of the tribits are not scrambled and the FEC encoder is a convolutional encoder. In alternative embodiments, the tribits on bus 360 can be pseudorandomly scrambled prior to being received by the FEC encoder 402. In other alternative embodiments, the FEC encoder can use block codes.

The purpose of the forward error correction encoder 402 is to add one or more redundant bits to each tribit so as to improve the error rate for the energy per bit-to-noise power density ratio resulting from the chosen modulation scheme. In the preferred embodiment, the FEC encoder 402 is a trellis encoder for a 16-QAM, Rate 3/4 trellis code having 16 states, a pi/4 rotational invariant, no parallel paths and an effective code length of 2. In yet another alternative embodiment, the forward error correction encoder 402 could be a Reed-Solomon Encoder which generates a first set of code words which are then further encoded in a trellis encoder. An advantage of using trellis encoded modulation either with or without Ree-Solomon coding is that it allows redundancy to be added to the payload data so as to enable forward error correction without increasing the symbol rate and the consumed bandwidth. Trellis encoded modulation uses redundant bits to map the payload data into a larger constellation of possible points (called signal space coding). The bandwidth required for transmission is not increased, nor is total noise admitted by the receive filter. Basically, trellis encoding uses a channel coder to receive each k payload bits and convert them into n bits where n is greater than k and includes some redundant bits which contain information about the k payload bits. The n bit group is then processed by a modified line coder to produce symbols for transmission from a constellation having size 2^n. Significant coding gains can be achieved in this way. In the invention, a coding gain of approximately 4 db is obtained. The main advantage of using trellis coded modulation is the ability to reduce the error rate or increase the number of payload bits without increasing the symbol rate and bandwidth consumed. This can be done using a constellation no greater than 2M. More details about trellis encoded modulation are contained in Lee and Messerschmit, *Digital Communication, 2d Ed.*, 1994 (Kluwer Academic Publishers, Boston), ISBN 0 7923 9391 0, which is hereby

incorporated by reference. Trellis encoded modulation is not required however to practice the invention, and, therefore, the encoders 402 and 526 in Figures 3 and 11, respectively, could be eliminated or replaced with simple encoders using any known error detection or correction encoding scheme and a mapper to map the resulting encoded symbols into points in a constellation.

In still other embodiments, forward error correction is not used, and the encoder 402 is an ARQ encoder which simply adds enough ECC bits to allow the receiver to detect an error and request a retransmission. The retransmission request is made on one of the command and control channels. In some block code embodiments, the forward error correction encoder 402 uses cyclic codes where the sum of any two code words is a code word and any cyclic shift of a code word is also a code word. Note that the Viterbi decoder 468 discussed below in the description of the receiver is used only when the forward error correction encoder 402 is a convolutional or trellis encoder.

Although the discussion of the forward error correction encoder 402 has not heretofore included any discussion of the modulation process carried out by modulator 410, Trellis-Coded Modulation (hereafter referred to as TCM) is preferred because of its lower error rate in the face of channel impairments. TCM modulation combines the forward error correction and modulation process by redefining the coding as the process of imposing certain patterns on the transmitted signal. This provide more effective utilization of band-limited channels as is the case for multiple access on HFC cable TV plants.

In the preferred embodiment, the forward error correction encoder 402 has multiple modes: which add different numbers of redundant bits while always maintaining the code word length at 4 bits. In a normal mode, one redundant bit is added per tribit. In a fallback mode when channel impairments are high, fewer payload bits are sent and more redundant bits are sent in each 4 bit code word.

The 4th bit in each tribit is part of the trellis modulation scheme and is generated by the convolutional encoder 402. A three bit constellation would normally have only 8 points. However, trellis modulation adds redundant bits interspersed in the information stream of tribits and increases the size of the constellation to enable more spacing between constellation points thereby enabling better discrimination between points by the receiver and lowering the bit error rate without increasing the bandwidth. In noisy environments like CATV media, trellis modulation is preferred, but some species of the invention will work without the redundant 4th bits and using a smaller constellation. The encoder, in the preferred embodiment, is a state machine but it could also be a lookup table implemented in RAM or ROM etc. The implementation of the state machine

is not critical as long as the implementation is fast enough to keep up with the chip clock data rate. For purposes of this discussion, it will be assumed that the convolutional encoder 402 is present.

M-ary Modulation in Code Division Multiple Access System

The output of the convolutional encoder 402 is an array of 4-bit digital numbers for each of symbols 1, 2 and 3 shown in Figure 4. Each of these 4-bit numbers has two bits representing a real part and two bits representing an imaginary part. Thus, the information vector [b] shown at 481 in Figure 7A for use in the matrix multiplication for CDMA spreading of each symbol is comprised of 144 4-bit elements. Each 4-bit symbol element in Figure 7A, such as element 483 represents one third of the information bits from the corresponding timeslot in the TDMA stream input the transceiver plus the redundant bit calculated by the convolutional encoder 402. Figure 7A illustrates how the information vector [b] for each symbol has its energy spread over time by the process of code division multiplexing implemented using matrix multiplication of the information vector [b] of each symbol times a matrix of orthogonal codes. The first two bits of each 4-bit symbol element are used to define the amplitude of either the I or Q coordinate, and the last two bits are used to define the amplitude of the other coordinate. The constellation of input point mappings of all possible points defined by a 4 bit symbol element or "chip" is shown in Figure 5. Figure 5 maps each of 16 possible input points, i.e., permutations of the 4 bits of each chip in each symbol array to a point in space defined by the in-phase or I axis for the real part and the quadrature or Q axis for the imaginary part of each point. The I coordinate of each point represents the amplitude for that point imposed upon the sine wave carrier fed to the modulator 410 in Figure 3 to modulate that point. The Q coordinate of each point in the constellation represents the amplitude imposed by modulator 410 on the cosine wave carrier fed to it in order to modulate the point in QAM trellis modulation. Figure 6 is a table listing all the possible 16 combinations of 4 bits in the Code column and the corresponding 2's complement digital representation of the real and imaginary coordinates for each combination in the Inphase and Quadrature columns, respectively. For example, the input point 1100 maps to a point having a +3 imaginary coordinate and a -1 real coordinate on the constellation of Figure 5. The mapping of Figure 5 was selected to give maximum separation between points in the constellation for best noise immunity, but any other mapping would also work. Likewise, 2's complement representation is not required for the coordinates as they can be represented in other number systems as well. The function of the trellis encoder 402 is to select the bit to append to each tribit to put it at a place in the 16 point constellation of Figure 5 which gives maximum noise

immunity. This selection is made according to known trellis modulation principles based upon the previous states. In other words, trellis encoder 402 and state memory 404 comprise a state machine which transitions to one of the 16 states or points in the constellation based during each chip time based upon the incoming tribit data and the previous states. The memory 404, in the preferred embodiment, is large enough to record the last state for each of the time slots, so as each tribit arrives, the last state for the time slot from which the tribit was generated is looked up in memory 404, and the tribit is encoded based upon that channel's prior state.

The stream of 4-bit symbol elements that are output from the encoder 402 are stored in memory 406 as three different linear arrays corresponding to symbols 1, 2 and 3 in Figure 4. Each 4-bit symbol element is a complex number comprised of 2 bits which define the I or inphase coordinate of a constellation point and 2 bits which define the Q or quadrature coordinate of the same constellation point. These two I and Q values are output on buses 1068A and 1070A in Figure 3.

After passing the tribit stream from the framer 400 through the encoder, the resulting 4-bit data streams are stored as separate I and Q information vector arrays for each symbol in memory 406 in Figure 3. The 144 array elements of each symbol define an information vector b for each symbol. The code division multiplexer 408 then spreads each information vector separately with a separate orthogonal code for each channel and combines the spread data into a single orthogonally coded data stream.

Figure 7A shows the matrix multiplication process which is performed within code division multiplexer 408 in Figure 3 to multiply each of the two linear arrays that define each symbol times the orthogonal code matrix [c] identified as matrix 407 in Figure 7A. In the preferred embodiment, the matrix multiplication is performed by a microprocessor, but any machine that can do the matrix multiplication will suffice to practice the invention.

The encoding in CDMA MUX 408 spreads the energy of the symbols over time using orthogonal codes or orthogonal, cyclic codes. This is done in two steps. First, a linear array information vector of just real parts, i.e., inphase coordinates of the symbol to be transmitted, symbolized by array 405 in Figure 7A, is multiplied by the code matrix 407. This operation generates another linear array of real or inphase coordinates along the R axis of a result space in a results constellation similar to the constellation of all possible input points shown in Figure 5. This first linear array 409 defines the real axis coordinates in the result constellation for a plurality of chips from the first symbol to be transmitted.

Second, the same process is repeated for the imaginary coordinate linear array (not shown) for the same symbol. This results in another linear array (not shown) comprising the imaginary or quadrature coordinates of the chips in the results array.

The real component array, represented by linear array 409, is part of an overall result or "chips out" array which contains both the real and imaginary coordinates of a plurality of chips to be transmitted. These chips map to points in the result space, and the points in the result space map to whatever points in the input point space that are defined by the real and imaginary components in the information vector array b, of which array 405 is the real part. The mapping between the input point space and the results space is defined by the contents of the code matrix and the orthogonal codes.

Before performing the matrix multiplication, the 2's complement values of the real and imaginary components of the information vector b input array are converted to their decimal equivalents as shown in Figure 7A in some embodiments. Figure 7A is a simplified version of the system in which there are only 4 channels resulting in 4 elements of each symbol. The 4 real components of the information vector b shown in array 405 after conversion to their decimal equivalents, are, respectively from top to bottom, +3 (first three bits of channel 1), -1 (first three bits of channel 2), -1 (first three bits of channel 3) and +3 (first three bits of channel 4). This column of numbers is multiplied by the first row in the code matrix to yield the result 4 as the first real component in array 409 of the results array. This result is derived from summing the partial products as follows [(3x1) + (-1x1) + (-1x1) + (3x1)] = 4. The next component down in the real part array 409, i.e., 0, is derived by multiplying the next real component down in the array 405 (-1) times the second row of the code matrix in a similar manner yielding [(-1x-1) + (-1x-1) + (-1x1) + (-1x1)] = 0. In the preferred embodiment, arrays 405 and 409 would be 144 elements long, and the code matrix 407 would have 144 elements in each row and would have 144 rows. The orthogonal codes are actually the columns of the array. Note that the channel 1 element always gets multiplied by an element of the first column and so on for all the elements of array 405 as array 405 is multipled by each of the 4 rows in array 407. Thus, the first column in array 407 is the orthogonal code used to spread out the bandwidth of the data from the channel 1 timeslot. For ease of generation, the set of orthogonal pseudorandom codes in matrix 407 is also cyclic.

Because each orthogonal code used in array 407 is also pseudorandom, and the rate of generation of the chips in the result vector (the chip rate) is much higher than the bandwidth of the input data represented by the information vector 405, the

bandwidth of the resulting signals defined by the result vectors generated by this process is spread into an extremely broad spectrum.

The CDMA MUX 408 in Figure 3 that does the matrix multiplication can be a programmed microprocessor or a dedicated custom logic circuit, etc. Any design which can perform the multiplication of the information vector times the code elements for all the active channels will suffice. Since the code matrix is comprised of purely 1's and -1's, the multiplication is made simpler. If the codes in the code matrix are Hadamard codes, the matrix multiplication can be made using the Fast Hadamard Transform algorithm in a digital signal processor or microprocessor. If the code matrix is comprised of sin and cosine terms, the Fast Fourier Transform can be used. Although any orthogonal or any cyclic code can be used to practice the invention, cyclic codes are preferred because they are easier to generate.

The resulting real and imaginary component linear arrays of the results or chips out array are stored in a memory within the CDMA Mux 408 which is not separately shown. The components of these two arrays are then output on separate I and Q buses to a modulator 410 where they are used to amplitude modulate the amplitudes of two RF carriers that are 90 degrees out of phase using a trellis modulation scheme. The resulting two AM carriers are summed and output on the transmission media 412. This is done as illustrated in Figure 8. Not shown in an up conversion or down conversion frequency translator to move the resulting signal in frequency to the band designated for use. The frequency band designated for use depends upon whether the transmission media 12 is a cable TV system, satellite system etc. and further depends upon whether the signals are travelling in the upstream or downstream direction.

Referring to Figure 8, more details of the coordination of the multiplexer 408 and the modulator 410 and the internal details of the modulator 410 in Figure 3 are illustrated for the transmitter modulators in either the RU or CU. Although there are slight differences between the RU and CU transceivers, they are generally the same, with some differences discussed elsewhere herein. The result or chips out array is stored in memory 411 which is part of the CDMA MUX, and comprises the real or inphase array 409 and the imaginary or quadrature array 413 of the 144 result points or chips in the result space. On every chip clock, one result point or chip comprising a real component and an imaginary component is output on bus 451 to a bit parsing unit or bit splitter 453. The bit parsing unit 453 splits off the real component and outputs those bits on bus 417. The imaginary component will be parsed out, and those bits will be output on bus 419.

Because the RF signals that carry the information from the 144 channels must share the transmission media with other RF signals having adjacent frequencies, two optional digital passband filters 421 and 423 are used to limit the bandwidth of the signals on buses 417 and 419 to avoid interference with signals on neighboring frequencies. The digital signals on buses 417 and 419, when converted to their decimal equivalents usually have rapid transitions between levels in adjacent intervals. This is illustrated in Figure 9 which is a plot of the changes in amplitude over time of the real components of the results vector for the array 409. These filters 421 and 423 are Nyquist passband filters having center frequencies at the carrier frequency and having 6 dB bandwidth points which are each separated in frequency from the center frequency by a frequency gap $1/(2T_c)$ where T_c is the chip rate period, i.e., the time between transitions from one chip level to the other. The Nyquist filters 421 and 423 remove high frequency Fourier components caused by sharp edges in such signals. This filtering effectively rounds off corners of the waveform defined by the transitions between successive chip levels in the "chips out" array and limits most of the power density in the Fourier spectrum of such signals to a 6 MHz band centered around the frequency of the RF carrier generated by local oscillator 425. This local oscillator 425 generates a sine wave, RF carrier at a frequency selected to be compatible with the switching rate of CDMA multiplexer 408 and to not interfere with existing cable TV service signals on adjacent frequencies. Since the local oscillators in the RUs and CU that are used for the modulators and demodulators all run synchronously locked in phase to each other and are kept in phase in the RUs by the carrier recovery circuits described elsewhere herein, the local oscillators that generate carriers will all be designated 425 even though they are separate circuits one of which is in the CU and some of which are in the RUs.

The local oscillator COS wave is applied to the carrier input 427 of an amplitude modulator 429 which also receives the filtered real component of each chip on bus 431. The modulator 429 modifies the amplitude of the carrier signal on line 427 in accordance with the amplitude of the decimal equivalent the real component on bus 431 and outputs the result on bus 443.

The imaginary or quadrature component of each chip, after filtering, is input on bus 433 to another amplitude modulator 435. This modulator receives at a carrier input 437 a sine wave of the same frequency as the cosine wave on line 427, but shifted in phase by 90 degrees by phase shifter 439. These local oscillator SIN and COS signals on lines 427 and 437 are actually generated in the carrier recovery circuit 515 in Figure 3 in an RU and are locked in frequency and phase to the pilot channel tone send

downstream from the CU during timeslot 0. Modulator 435 modifies the amplitude of the sine wave in accordance with the amplitude of the imaginary component on bus 433, and outputs the result on line 441. Lines 441 and 443 are coupled to a summer 445 which sums the two waveforms and outputs them on the shared transmission media via line 412.

In some embodiments, the line 412 may be coupled to suitable interface circuitry to drive the signal on line 412 into a wireless or cellular system, a terrestial microwave link, a coaxial cable of a cable TV, telephone or other system, a fiber optic link of a cable TV, telephone or other system, a local area or wide area network or any other media developed in the future for real time communication of data. Such interface circuitry is known and will not be described further herein.

In the preferred embodiment for purposes of carrier recovery by the RUs for downstream data, the signal from the local oscillator 425 in the CU transmitter modulator is also provided as pilot channel data on line 501 to a command and control buffer 503 in Figure 3 . The command and control buffer stores data to be transmitted on the command and control channels for system management, contention resolution, ranging etc by either the RU or CU transceiver. This data is received from the CPU 405 via bus 497. Bus 505 couples this data to an input of a switch 507 which has a second input coupled to receive the payload data on bus 360 from the framer. The switch selects one of these buses as the source of data which is output on bus 509 to the forward error correction encoder 402 for trellis encoding. Switching of switch 507 is controlled by CPU 405 by a control signal on line 511.

Pilot Channel data bus 501 is shown in phantom in Figure 3 to represent the fact that this pilot channel data is, in the preferred embodiment, only input to the command and control buffer 503 if the transceiver of Figure 3 is in the CU.

If the transceiver of Figure 3 is in the RU, then no pilot channel data is input to the command and control buffer. Instead the local oscillators in the receiver and transmitter are synchronized to the frequency and phase of the pilot channel. Carrier recovery, i.e., carrier synchronization of the frequency and phase of the RU local oscillator 425 to the pilot channel signal broadcast in timeslot 0 from the CU is the function of carrier recovery circuit 515 in Figure 3. Specifically, a local oscillator carrier signal is provided by a carrier recovery circuit 515 to demodulator 460 as the COS signal on line 427. The local oscillator COS signal on line 427 is synchronized in frequency and phase with the CU local oscillator carrier signal which was used to modulate the signals received at input 521. Likewise, in the RU transmitter 401 of Figure 3 the carrier recovery circuit 515 transmits a local oscillator signal on line

427 which is synchronized in frequency and phase to the pilot channel signal. This signal is input to the RU transmitter modulator 410 so that its signals will be coherent to the CU receiver. However, preamble data must be inserted into every timeslot's data for use by the CU receiver to acquire the phase and amplitude of the signals for that timeslot. This is because every RU is at a different distance from the CU so even though the RU transmitter modulators use the same frequency and phase local oscillators as the CU, the differing propagation times and channel impairments cause phase and amplitude ambiguity which the CU must resolve separately for each timeslot.

The carrier recovery circuit 515 can be any conventional phase-locked loop clock recovery circuit, Mth power loop, Costas loop, suppressed carrier-tracking loop, etc..

Referring again to Figure 3, the apparatus and method by which upstream carrier recovery, gain control and symbol synchronization is achieved will be described. Even though all RU local oscillators are synchronized in frequency and phase with the pilot tone from the CU, the differing distances from each RU to the CU cause two different problems. The QAM signal demodulation used in the preferred embodiment depends for its accuracy on the ability to accurately distinguish between the amplitudes and phases of each received point. The differing propagation times and differing channel impairments experienced by each RU's signal, cause both amplitude and phase errors in the received data that must be determined and corrected for to obtain accurate QAM demodulation at the CU receiver. The way this is done is for each RU to send known preamble data to the CU in the timeslots currently assigned to that RU before the block of payload data is sent. The CPU in the CU assigns the timeslots to the various RUs and so informs them in management and control messages on the management and control channels. In the embodiment shown in Figure 3, the CPUs in the RUs keeps track of and help control the process of breaking the payload data from their peripherals/user devices into 8 bit bytes, adding a 9th bit to support the higher level protocol and sending the 9-bit bytes during the assigned timeslots. Before the payload data is sent however, the CPU in the RU activates a Preamble signal on line 1074 which controls switching by a multiplexer 1076. This multiplexer receives the encoded I and Q information vector data on buses 1068A and 1070A at one input and predetermined, fixed I and Q values for preamble data on buses 1078 and 1080 at another input. When the switching control signal on line 1074 is activated, multiplexer selects the data on buses 1078 and 1080 for coupling to buses 1068B and 1070B for storage in memory 406. The data on buses 1078 and 1080 define a known point 3-j in the QAM constellation.

Line 1074, buses 1078 and 1080 and multiplexer 1076 are only present in the RU transmitters since the technique described here is used only in the upstream data to achieve proper synchronization.

In the CU receiver, the slicer detector 466 is responsible for comparing the received data to the known preamble constellation point during preamble reception to determine the gain and phase errors. The received signal takes the form:

$a*e^{j\varnothing}*s(t)$

where s(t) is the desired signal;

a = the amplitude error caused by channel impairments and the near-far problem; and

$e^{j\varnothing}$ = the phase error caused by channel impairments and the near-far problem.

The slicer detector 466 in Figure 3 encompasses several circuits shown in Figure 12. The slicer detector 466 operates to derive a multiplication factor to multiply times the received signal so as to cancel the amplitude and phase error such that s(t) is detected as the constellation point 3-j without any slicer error. The amplitude and phase error coefficients in the multiplication factor which reduce the slicer error to 0 are then stored in memory 796 in Figure 3 for use by the slicer in receiving the payload data for that timeslot(s) assigned to the RU for which the multiplication factor was stored.

Upstream Carrier Recovery Error Correction Factor Per Timeslot

Specifically, the job of the CU receiver slicer detector 466 is to determine the correct 1/a and $e^{-j\varnothing}$ coefficients in a multiplication factor of the form:

(5) $(1/a)*e^{-j\varnothing}$

where 1/a is the gain correction coefficient to solve the near-far problem and correct for channel impairments; and

$e^{-j\varnothing}$ is the phase error correction coefficient to solve the near-far problem and correct for channel impairments.

The near-far problem involves interference with reception of weak signals transmitted from a remote transceiver by strong signals transmitted by a near receiver. In the prior art, this is often solved by time division multiplexing so that the two transmitters are never transmitting at the same time. In the environment of the invention, this solution will not work since all RUs have to be able to transmit whenever they need to transmit if bandwidth is available. Therefore, in the invention, the amplitude levels of the signals transmitted by the RUs are controlled so that all signals arriving from the RUs at the CU should arrive at approximately the same amplitudes, and channel impairment effects are corrected by gain level adjustments in the CU

receiver at a point before the baseband signal enters the slicer so as to minimize interpretation errors caused by amplitude errors. Likewise, a rotational amplifier in the slicer detector corrects for phase errors caused by the differing propagation delays and channel impairments prior to the baseband signal entering the slicer to minimize this source of errors. For a discussion of the iterative process carried out by this circuitry during the preamble for each timeslot to establish the values for the amplitude and phase error correction coefficients for use in receiving the payload data for that timeslot, see the discussion of the cooperation of G2 amplifier 788, rotational amplifier 765, slicer 800, control loop 781 and memory 796 in Figure 12.

Thus coherent modulation and detection is used for both upstream and downstream transmissions. For quadrature modulation schemes, the modulator 410 and the demodulator 460 includes phase shift circuitry to shift the phase of the signals on line 427 by 90 degrees so that both sine and cosine local carrier waveforms which are synchronized in phase and frequency to sine and cosine waveforms used in the CU modulator are available for the modulation and demodulation tasks in the RU.

The pilot channel data on timeslot 0 is spread with a dedicated CDMA code in CDMA multiplexer 408 for transmission on the timeslot 0 management and control channel as the pilot channel data which encodes the CU master clock. Each RU receiver includes carrier recovery circuitry 515 which monitors this pilot channel signal and generates the synchronization information on line 427.

The form of carrier recovery described above is preferred for coherent detection. In alternative embodiments, incoherent detection could also be used using any of the well known incoherent detection apparatus. Such incoherent receiver technology is described in Haykin, *Communication Systems*, at page 503-505 and is hereby specifically incorporated by reference herein.

Another form of synchronization is symbol synchronization. The receiver must know the instants in time when the modulation can change its states. That is, the RU and CU receivers must know the start time and finish time of each chip in order to decipher what that chip was. This allows the receiver to determine when to sample and when to quench its product integrator or other chip state detection circuitry for purposes of starting the chip decoding process. Symbol synchronization in the context of the invention is recovery of the CU chip clock in each RU. In the preferred embodiment, recovery of the CU chip clock is done by correlating in each RU a known barker code transmitted during every gap by the CU, with the barker code encoding the chip clock therein. Each RU uses a correlator with an early-late gate to detect the barker code and get the RU's chip clock synchronized with the CU chip clock encoded in the barker code.

This process of chip clock synchronization is carried out by the frame detector 513 in Figure 3 and frame detector 882 in Figure 12. The frame detector 513 and the frame detector 882 each includes both coarse and fine tuning circuitry. The coarse tuning circuitry performs downstream frame synchronization by locating the gap in each CU frame transmission by finding a known barker code transmitted by the CU in the gap. Once the gap is located, the time base circuit 886 in Figure 3 is synchronized to this receive frame timing reference by a signal on bus 1031, and time base 886 in Figure 12 is synchronized by frame detector 882 to the receive frame timing reference by a signal on line 1092. The time base circuit 886 is comprised of a series of cascaded counter stages that receive a high speed input clock that is phased locked by the clock steering signal from the frame detector (line 900 in Figure 3, line 192 in Figure 12). The cascaded counters generate the chip clock, frame clock, superframe clock and kiloframe clock signals. The timebase in both the receiver and transmitter of each modem includes a chip counter and a frame counter as well as sampling registers which are used to correctly align the timebase with external signals. Once the time base is aligned to these external signals, all internal timing needs of the modems are served by the time bases so that they do not depend upon external signals for operation, but the external signals are monitored for loss or shift. In the case of the CU, the external signals to which the time base is aligned are the time division multiplexed inputs to the transmitter. In the case of the RU, the external signals are the gap detect Frame and Kiloframe signals derived from the downstream data.

The time base circuit 886 provides these signals which include receive frame timing reference information to any circuit in the receiver or transmitter that needs this information such as the receiver's orthogonal demultiplexer 462 in Figure 3 and the orthogonal code demultiplexer 766 in the receiver of Figure 12. The time base circuit also continually checks the position of the gap by sampling a gap detect signal from the frame detector over multiple frames so as maintain frame synchronization and know when frame synchronization has been lost. When the gap position is lost, the modem immediately attempts to resynchronize to the gap.

The orthogonal code multiplexers in the RU and CU transmitters also get frame timing reference signals, but these frame timing reference signals establish the boundaries of the CU's frame timing reference since each RU transmitter times its transmissions and other processing so that frames transmitted therefrom arrive at the CU coincident with the CU frame boundaries. And of course the CU transmitter needs to transmit its frames in synchronism with the CU frame boundaries. To that end, the receive frame timing reference signal generated by the frame detector 882 in Figure 12

and 513 in Figure 3 is sent to the modem's local CPU or other control circuit 405 via bidirectional bus 902 in Figure 3 and via bus 883 and DMA memory 763 in Figure 12. The CPU or other control circuit 405 then uses this frame timing reference to set the timing of the transmit frame timing delay T_d on line 499 to the transmitter frame circuits 400 and 508 in Figure 3 and 11, respectively.

The fine tuning circuitry in the frame detectors 513 and 882, in Figures 3 and 12, respectively, performs clock recovery for symbol synchronization by using early-late gating techniques in conjunction with correlation to generate a clock steering tracking error signal on bus 900. This signal corrects the phase of a voltage controlled oscillator 784 in Figure 12 which is used by time base generator 886 in the RU to generate a local chip clock signal which is synchronous with the chip clock in the CU.

The coarse tuning circuitry in the frame detectors 513 and 882 cooperates with a software process running in CPU 405 to locate the CU frame gaps. This is done using control and timing signals on bus 902 on the CPU and the real and imaginary data components on bus 904 output by the demodulator 460 in Figure 3 and the matched filter 761 in Figure 12. This gap location process is accomplished by continually moving the boundary of a sliding correlation window until a correlation peak appears at the same time at least twice consecutively. How this works will be explained in more detail with reference to Figure 13.

Referring to Figure 13, there is shown a block diagram of the preferred form of a ranging detector which forms the heart of frame detector 513 in each RU and is used in the CU for ranging detection of barker codes. Hereafter, the circuit of Figure 13 will be referred to as the ranging detector even though it has frame detection and chip clock synchronization functions as well.

The ranging detector has an acquisition mode and a tracking mode. In acquisition mode, it is simply trying to rapidly find a known barker code arriving in the collection of signals on bus 904. In the preferred embodiment, where the transmit data is passed through a raised squared cosine filter, bus 904 is coupled to the output of a matched filter like matched filter 761 in Figure 12 having a transfer function which is the inverse of a raised squared cosine function, but in other embodiments, these two filters may be eliminated. Bus 904 carries data defining the real part of the received signal on lines 906 and the imaginary or quadrature part of the received signal on lines 908.

In acquisition mode, the interest is in quickly finding the gap by correlating the incoming signals with the known barker code, but this can be done by simply looking at the sequence of signs of signals received since the known barker code is a known, unique

sequence of chips of differing signs but constant amplitude. The barker code can be located effectively in tracking mode by looking at only the sequence of differing signs in the received data. Therefore, in tracking mode, the CPU sends selection control signal **acq** on bus 902 to control the state of switches 906 and 908 so as to select the signals on buses 910 and 912. The signals on buses 910 and 912 are the outputs of circuits 914 and 916 which serve to compare the incoming signals on bus 904 to zero and output a first number if the sign of the incoming chip is + and output a second number if the sign of the incoming chip is -. When **acq** is not asserted, the raw data on buses 918 and 920 is selected for passing through switches 906 and 908. The **acq** signal also passes through OR gate 922 to gate the output signals from switches 906 and 908 through to finite impulse response filters 924 and 926 in acquisition mode for correlation. The OR gate 922 also receives a GAP_a signal which is asserted by the CPU via bus 902 when the CPU thinks it is in the gap by virtue of signals from the frame detector. Therefore, the signals on buses 928 and 930 from switches 906 and 908 will be correlated by FIR filters all the time when the ranging detector is in acquisition mode and, while in tracking mode, only during the gap.

The FIR filters 924 and 926 have impulse response functions which are programmable and are set by the CPU 405 to match the barker sequence which the receiver is looking for. The barker sequence being sought is defined by data written by CPU 405 into register 932. When this exact sequence of + and - chips resides in either one of the FIR filters, the filter output will peak. Absolute value circuits 934 and 936 are coupled to the outputs of the FIR filters, and output the absolute values of the FIR output signals on buses 938 and 940. Circuit 942 has two different modes which are selected by the **acq** signal on line 943. In acquisition mode when the receiver is trying to initially locate the gap, circuit 942 selects the greater of the signals on buses 946 or 948 for output on bus 944. In tracking mode, the sum of the signals on buses 946 and 948 is output on bus 944.

Comparator 950 acts to set a minimum threshold above which the FIR output peaks must rise before they are counted as possible reception of the CU barker code. Comparator 950 compares the signals on bus 944 to a threshold level on bus 945, and, if the threshold is exceeded, outputs a logic 1 on bus 951 during the interval when the threshold is exceeded. The threshold level is set by data written into register 952 by CPU 405 via bus 902. The number of peaks is counted by a false alarm counter 952 the output of which is stored in register 960 which is periodically read by the CPU in a process of monitoring and controlling the ranging detector. A process in CPU 405 which monitors the number of false alarms, sets the number of frames over which false alarms

will be counted by writing a number of frames into register 956. This number is loaded into interval counter 954 which counts down from that number by counting the GAP_b signals on line 957 which occur one per frame. When the count reaches zero, line 958 is activated which clears the false alarm counter 952, strobes the count before clearing into register 960 and reloads counter 954 from register 956. When the CPU determines that the number of false peaks is too large according to the number in register 960, it raises the threshold by writing new data to register 952 to raise the threshold.

Course tuning to find the gap is accomplished by the ranging detector as follows. The CPU starts with an estimate of when it thinks the gap will start. At that time, signal GAP_a on bus 902 is asserted during each frame interval. The CPU only wants to look at peaks during the gap in each frame interval, so it uses a sliding window to restrict the time during which it is looking for peaks. The sliding window is symbolized by bracket 962 in Figure 14. The boundaries of this window are established by data written by CPU 405 to register 964 in a manner to be described below.

Circuit 970 passes only the first peak on the output of the AND gate 968 which occurs after the GAP_a signal indicates the gap is thought to have started. A time base counter 972 counts chip clock signals on line 974 and is cleared by the GAP_a signal every frame. When circuit 970 passes a peak (actually a logic 1 level) through on bus 976, the current count of the time base counter 972 output on bus 980 is sampled and stored in register 978. The count value on bus 980 is also coupled to a comparison input of a greater than or equal to comparator 965, the other input of which is coupled to receive the output of the register 964. The output of the comparator 965 is the gating signal on line 966. Since the count of time base counter 972 will be reset to 0 at the moment the CPU thinks the gap is starting, the count stored in register 978 represents an offset error indicating how much later the gap may have actually started compared to the time the CPU thought the gap was starting.

Figure 14 is a timing diagram that helps explain the course tuning process to find the time the CU frame gap occurs which is carried out by the RU receivers. Timeline A of Figure 14 represents the initial sliding window position 962 set by the CPU during a first frame before it is sure where the gap is and shows the times of two peaks observed during frame 1. Timeline B represents the position of the sliding window and the peaks observed during frame 2. Initially, the CPU does not know where the gap is, so the software process decides to watch for peaks on line 976 for the whole frame. Accordingly, the CPU writes a 0 into register 964 at time T0 and simultaneously activates the GAP_a signal. Activation of the GAP_a signal resets the timebase counter

972 and drives a logic 0 onto bus 980. The 0 in register 964 is compared to the 0 on bus 980 by greater than or equal to comparator 965 which finds an equality and sets line 966 to logic 1 thereby gating pulses on bus 951 from the threshold comparator through to the first pulse selection circuit 970. Comparator 965 drives line 966 to logic 1 anytime the number on bus 980 is greater than or equal to the output of register 964. This action opens sliding pulse observation window 962 in Figure 14 at time T0. The window will remain open until the end of the frame.

During frame 1, shown on timeline A of Figure 14, a noise pulse 990 is gated through circuit 970 at time T1, and the actual barker code pulse 992A which occurs at time T7 is blocked by circuit 970. The occurrence of noise pulse 990 causes sampling of the count on bus 980 by the register 978, which is indicated in Figure 14 as sample 1 at time T1. This value is read by the gap acquisition process executing on CPU 405 and stored for later comparison.

Because the noise pulse 990 was random, it does not occur again at time T1 in the second frame shown on timeline B of Figure 14. Instead, another noise pulse 994 occurs at time T3, later than T1, and another barker code pulse 992B occurs at time T7. First pulse selection circuit again gates pulse 994 through and blocks pulse 992B. This causes the taking of sample 2 of the count on bus 980 during frame 2. The coarse tuning gap acquisition process reads the value stored in register 978 and compares this value to the value previously read from this register during frame 1. The CPU concludes pulse 990 occurred at a different time than pulse 994, and, therefore, pulse 990 was noise and cannot be attributed to the barker code because if it were the barker code, it would not be random and would have occurred at the same time. Accordingly, the gap acquisition process moves the position of the window 962 for frame 3 to open at a time just before the occurrence of pulse 994 so as to eliminate any pulses before that time from consideration but so as to analyze pulse 994 to see if it is attributable to the barker code. The CPU gap acquisition process moves the position of window 962 by taking the sample 2 number from register 978, subtracting a fixed amount from it, and writing the result to register 964.

The situation for frame 3 is shown on timeline C of Figure 14. The window 962 opens at time T2, but because pulse 994 in frame 2 was noise, it does not occur again in frame 3 at time T3. Instead, noise pulse 996 occurs at time T5, and is gated through by circuit 970 while the actual barker code pulse 992C is blocked. Pulse 996 causes sample 3 to be taken. The gap acquisition process compares sample 3 to sample 2 and concludes that pulse 994 was noise because pulse 996 did not occur at the same relative time (relative to the occurrence of GAP_a). Accordingly, the gap acquisition process

concludes that the window 962 can be moved again. This time, the window is moved to open at a time T4 just before the time of occurrence of pulse 996 at time T5.

During frame 4, window 962 opens at time T4, but no pulse occurs again at relative time T5, but the barker code pulse 992D occurs again at time T7. This barker code pulse is gated through by circuit 970 and causes sample 4 to be taken. The gap acquisition process reads sample 4 and compares it to sample 3, and decides that pulse 996 was noise because pulse 992D did not occur at the same relative time. Accordingly, the gap acquisition process moves the position of window 962 again so as to open at a time T6 just before the occurrence of pulse 992D.

The situation during frame 5 is shown on timeline E of Figure 14. The window opens at time T6 thereby precluding consideration of any pulses occurring before T6. Another barker code pulse 992E occurs again at relative time T7 which is gated through as the first pulse in this frame after the window opened by circuit 970. This causes the taking of sample 5 which the gap acquisition process compares to sample 4 and concludes that the relative times of occurrence of pulses 992D and 992E were the same. The gap acquisition process then concludes that pulses 992D and 992E were barker code pulses and that it has found the gap. Accordingly, the gap acquisition process leaves the window 962 set to open at time T6 in frame 6 shown on timeline F of Figure 14 thereby ignoring noise pulses 998 and 1000 which occur before T6. The gap acquisition process then moves the time of activation of GAP_a to time T7, as shown on timeline G in Figure 14, and switches the ranging detector to go into tracking mode for the chip clock recovery process by deasserting the **acq** signal on bus 902.

The chip clock recovery process is carried out by early-late gate sampling circuitry in Figure 13 and, in the preferred embodiment, begins after the gap acquisition process. The basic concept is illustrated in Figure 15 which is a diagram of the sampling by the early-late gating circuitry of the output of the FIR filters (correlator output) when phase lock with the chip clock has been achieved. Curve 1002 represents the output signal on bus 944 from the correlation process that occurs in the FIR filters 924 and 926 between the known barker code (defined by coefficients in register 932) and the incoming signal. The major peak 1004 centered on time T0 (a different T0 than in Figure 14) represents the correlator output when the barker code sent in the gap by the CU arrives and is perfectly aligned in the FIR filters 924 and 926 with the data in the register 932. This register contains data defining the + and - polarity sequence of the individual elements of the barker code sent by the CU. Every CT-2 chip clock (8 chip clocks), a new digital sample of the received signal enters the FIR filters. The FIR filters do a summation of the results of each stage every CT-2 chip

clock. When all the samples of the barker code have entered the FIR and are aligned with the + and - polarity sequence that defines the barker code the receiver is looking for, the summation on the CT-2 chip clock that results in the alignment causes the peak 1004 at the output on line 944. Peaks 1006 and 1008 are examples of the summation results in the FIR filter before and after perfect alignment occurs. Points 1010 and 1012 represent sample points each of which is spaced apart from time T0 by one CT-2 chip clock. When the local oscillator is exactly aligned in phase with the phase of the local oscillator signal generated by the CU local carrier oscillator 425, the amplitudes of sample points at 1010 and 1012 will be the same. When there is some phase error, the two sample point 1010 and 1012 will have unequal amplitudes because pulse 1004 will not be symmetrically centered on T0. This generates the track error signal on line 900 in Figure 13 which causes the phase of a chip clock voltage controlled oscillator in a phase locked loop (not shown) to shift in such a manner as to alter the timing in which the data samples are fed into the FIR filters 924 and 926 so as to get the correlator main pulse 1004 to center on time T0.

The manner in which this clock recovery process is carried out by the circuitry of Figure 13 is as follows. Circuits 1014 and 1016 are the digital equivalents of sample and hold circuits. Circuits 1018 and 1020 are each delay circuits that each impose a CT-2 chip clock delay on a sample signal on line 1022. This sample signal is generated by the CPU 405 once per frame at a predetermined time in the gap after the GAP_a signal is activated. The sample signal cause circuit 1014 to sample the magnitude of the pulse 1004 on line 944 so as to take sample 1010 in Figure 13. This sample value is coupled to one input of a subtractor 1024, the other input of which is the magnitude of the signal on bus 944 (all processing is digital in the preferred embodiment). The subtractor 1024 constantly subtracts the first sample value 1010 stored in register 1014 from the changing values on bus 944 and presents the difference on bus 1026. Two CT-2 chip clocks later, the sample signal on line 1022 reaches register 1016 and causes it to store the difference value at that time on bus 1026. The value stored in register 1016 is the difference in amplitude between samples 1010 and 1012 in Figure 15. This value is the track error signal on bus 900. The track error signal is digitally integrated in phase lock loop 1030 in Figure 3 and the result is used as an error signal to correct the phase of a voltage controlled oscillator in PLL 1030 which serves to generate the local chip clock reference signal. This chip clock reference signal is coupled on bus 1032 to time base 886 which generates the timing signals needed to synchronize operations of the receiver and transmitter in Figure 3.

The ranging detector of Figure 13 also includes circuitry to determine when a barker code is exactly centered in the gap. This capability is used in the CU version of the ranging detector during the fine tuning process at the end of the ranging process where the CU sends instructions to the RU on how to adjust its transmit frame timing delay to exactly center its barker code in the gap. How this is done will be explained with reference to Figure 16 which illustrates the 3 permissible patterns of data at the output of comparator 950 for a centered barker code condition to be declared. Basically, the gap is 32 chip clocks wide, and is represented by window 1034. Comparator 950 will output 32 logic 0s or 1s during the gap interval, and these are shifted into shift register 1036. Two latches 1038 and 1040, each 16 bits wide, have their inputs coupled to the 32 bit parallel output bus 1042 of the shift register. These two registers 1038 and 1040 are constantly enabled, and are loaded with the contents on bus 1042 at the end of the gap with one taking the lower 16 bits and the other taking the upper 16 bits. For the barker code to be centered only the three bit patterns shown in Figure 16 are permissible. The first bit pattern on line A indicates two logic 1s on either side of the gap centerline 1044 and represents the data pattern that will be present in latches 1038 and 1040 when the RU's transmitted barker code has been exactly centered. The bit patterns on lines B and C represent acceptable conditions where the barker code is not exactly centered. The data patterns in registers 1038 and 1040 are read by the ranging process in execution on CPU 405 during the fine tuning process to deduce what instructions to give the RU to change its transmit frame timing delay T_d so as to move its barker code toward the center of the gap.

Returning to the consideration of Figure 3, the remaining receiver side circuitry of the transceiver will be described in more detail. As is the case with the transmit channel, the processing performed in the receiver may be performed using analog or digital or some combination of analog and digital circuitry. The receiver will be described as if all processing was digital as it is in the preferred embodiment. The signal received from the shared transmission media is passed through an analog-to-digital converter (not shown) and the resulting digital data stream is passed to a demodulator 460.

Figure 10 is a more detailed diagram of the structure of the demodulator 460 in the receiver. The received analog signal from the shared transmission media is coupled on line 461 to the analog input of an A/D converter 463. The stream of digital data resulting from the analog-to-digital conversion is simultaneously fed to two multipliers 465 and 467. Multiplier 465 receives as its other input on line 481, a stream of

digital values that define a local carrier sine wave having the same frequency and synchronous in phase with the RF carrier sine wave on line 437 in Figure 8. Multiplier 467 receives as its other input on line 427, a cosine signal generated by local oscillator 425 having the same frequency and synchronous in phase with the CU's broadcasted pilot channel broadcast in timeslot 0 which is the RF carrier cosine wave on line 427 in Figure 8. The inputs labelled SIN and COS in Figure 10 are generated by local oscillator 425 which is synchronized in frequency and phase with the pilot channel by the carrier recovery circuit 515 in Figure 3. A 90 degrees phase shift is applied to the local oscillator COS output to generate the SIN signal. The pilot channel signal is broadcast on one of the management and control channels (timeslot 0), and one of the CDMA codes is dedicated solely to this channel. This dedicated code is used to spread the pilot channel signal using conventional spread spectrum techniques. Each receiver decodes the pilot channel using this same code to recover the pilot channel carrier signal and applies the recovered signal to a phase detector in a phase lock loop which is used as a local oscillator source for the demodulator in each RU receiver section and the modulator in the RU transmitter section.

The results output from the demodulator on lines 469 and 471 are digital data streams which basically defines the mix products comprised of a fundamental carrier frequency and upper and lower sidebands. Digital filters 473 and 475 filter out the desired sidebands that contain the real and imaginary parts of each chip or result point that was transmitted. The stream of quadrature or imaginary components of the received chips are output on bus 477. The stream of inphase or real components of the received chips are output on bus 479. The receiver of Figure 3 also includes conventional phase lock loop circuitry for clock recovery and carrier recovery. In other words, the receiver recovers the bit clock timing used by the CU and synchronizes to it using conventional phase lock loop circuitry and also recovers and synchronizes to the sine and cosine carriers used by the CU to transmit the symbol data. These clock and carrier signals are then used for transmissions by the RU to the CU so that the CU can coherently communicate with the RU's without having to synchronize to different clock and carrier signals used by the RU's. In alternative embodiments, the RUs can use their own clock and carrier signals which are unrelated to the CU's versions and the CU can contain its own phase lock loop circuitry to recover these signals and synchronize to them in order to demodulate and interpret the data transmitted by the RUs.

In some embodiments, the streams of real and imaginary components of the 144 chips of each symbol on buses 477 and 479 are stored in two linear arrays in CDMA Demultiplexer 462 in Figure 3. The CDMA Demultiplexer 462 multiplies each of the

real and imaginary component arrays times the transpose of the code matrix used by the CDMA MUX 408 of whatever RU or CU that transmitted the data to reverse the orthogonal code encoding process. This matrix multiplication process results in two linear arrays of decoded chip real and imaginary parts for each symbol. These arrays are stored by the CDMA Demultiplexer 462 in memory 464. In alternative embodiments, the CDMA Demultiplexer processes the two streams of real and imaginary components "on the fly" such that they do not have to be first stored as input arrays in a memory in the CDMA Demultiplexer 462.

After the linear arrays of real and imaginary components for a symbol are stored in memory 464, the result for each symbol is an array of received chip points in a received chip space having a real axis and an imaginary axis. The mapping by orthogonal code transformation from the constellation of possible input points shown in Figure 5 leads to a constellation of possible points in a received chip space. A detector 466 examines the points in each of the arrays and compares the received chip points they define against the legitimate possible points in the received chip space. The detector, otherwise known as a slicer, is a known type of circuit and no further details are necessary herein. The function of the detector is to restore the gain and phase of the received signal, recover the chip clock therefrom and lock onto it so as to be in synchronization with the transmitter, determine the boundaries of each chip and determine the values for the I and Q coordinates of each received chip and compare the I and Q coordinates of each received chip point against the closest points in the constellation of legitimate possible points in the received chip space that could have been transmitted. The detector also helps the carrier recovery circuit 515 lock by generation of the Slicer Error signal on line 517 to lock the frequency of its local oscillators in the detector generating the sine and cosine signals used for demodulation to the phase and frequency of the sine and cosine carriers encoded in the data. The detector then makes a preliminary decision as to which of the possible legitimate points in the received chip constellation each received chip is likely to be.

The detector 466 then outputs its preliminary determinations to a Viterbi Decoder 468 which performs the prior art Viterbi algorithm. The Viterbi Decoder uses the 4th bit in each chip of each symbol to detect and correct errors. This is done by performing the Viterbi algorithm to derive the most probable tribit path defined by the points actually sent from the path in the received chip space defined by the 4-bit components of the symbols actually received, after they have been processed by the detector. These sequences of chips map a path through the space previously defined which is farther from the same type path mapped through a group of successive 8 point

constellations if only the tribits during each symbol time were plotted with no redundant bit added to each tribit. The fact that the chip path is farther from the 3 bit path makes it easier for the receiver to divine from the noise corrupted received data what the actual tribits transmitted were. Viterbi Decoder based systems are used by Qualcomm, Inc. in San Diego in cellular phone systems to combat noise in digital cellular phone transmissions, and the details of their patents and products are hereby incorporated by reference.

The output data points from the Viterbi Decoder are a stream of tribits. These tribits are stored in a memory in a deframer circuit 470 which functions to reassemble a replica of the TDMA data stream in the time domain from the incoming stream of chips or tribits comprising each symbol.

Preferred RU Transmitter Block Diagram

Referring to Figure 11, there is shown a block diagram of the preferred species of transmitter circuitry within the genus of the invention. The transmitter of Figure 11 is used in the transceivers of the RU modems. The CU transmitters are identical except there is no need for the access control circuitry 540 or the multiplexer 544.

In Figure 11, block 506 is the diversity code shuffler that implements the time to code transformation. The code shuffler receives a pseudorandom seed number on bus 499 which controls the pseudorandom order of shuffling of codes such that the various timeslots or channels are not always encoded with the same CDMA codes. Bus 499 also carries Tss data which defines which timeslots are assigned to this RU transmitter and an RU/CU signal which tells the code shuffler whether it is operating in an RU or CU. The R1 data on bus 499 defines reserved codes which cannot be used, and the T_d data is received from the CPU and receiver frame detector circuitry to set the transmit frame timing delay value for this RU so as to hit the gap with its barker code thereby achieving frame synchronization.

Block 508 is the framer circuitry that implements the variable transmit frame timing delays needed to implement the ranging process to achieve the necessary frame synchronization and time alignment of the CDMA spread channel data for synchronous CDMA. Block 548 is a buffer that stores the shuffled 4 bit groups of symbol elements which serve as the information vector [b] for the matrix multiplication performed by the CDMA Multiplexer 527. Code diversity is implemented by block 506 by controlling the order of tribits read for each symbol from framer memory 508 via read pointers sent to the framer on bus 532. The tribits exit the framer on bus 518 in the order dictated by the addresses on bus 532. They are pseudorandomly scrambled by scrambler

524 in the manner described below (in the preferred embodiment) and redundant bits are added by encoder 526 if operating in normal or fallback mode. Encoder 526 adds at least one bit to every tribit in the preferred embodiment to implement Trellis modulation. Some embodiments have no encoder, and some embodiments have an encoder which has no idle and/or no fallback mode.

The encoded bits are divided into real (or inphase) and imaginary groups by dividing each encoded tribit in half and outputting the first 2 bits as the real bits on bus 517r and the last two bits on bus 517i. Buses 517r and 517i are coupled to a switching circuit 544 which also receives as inputs real and imaginary components of access channel information on buses 542r and 542i. During normal payload transmission operations, switching circuit 544 selects the data on buses 517r and 517i for coupling on buses 546r and 546i to buffer memory 548. During access channel operations, (preamble transmission) switching circuit 544, under control of microprocessor 405, selects the data on buses 542r and 542i for coupling on buses 546r and 546i, respectively. The real and imaginary components in each tribit on buses 546r and 546i are written into buffer 548 in the order dictated by write addresses on bus 533. Elsewhere herein, the manner in which the multiplexer 544 is operated to overlay media access control data on buses 542r and 542i with payload data on buses 517r and 517i in buffer 548 is described. Buffer 548, when fully written, during each symbol time has 144 4-bit elements comprising an information vector the order of which is randomly scrambled anew each symbol time in the preferred embodiment. In other embodiments, the codes may be assigned sequentially during each symbol for all active timeslots, or a rolling sequential assignment of codes to all active timeslots may be used.

Returning to the consideration of Figure 11, the buffer memory 548 outputs two information vectors on buses 549r and 549i. The elements in these information vectors are, respectively, the first two bits in every Trellis encoded tribit as the real information vector and the last two bits of every Trellis encoded tribit as the imaginary information vector.

In Figure 11, block 510 generates the ranging barker codes needed for the ranging process to achieve frame synchronization. Preferably, this ranging barker code generator 510 is a state machine. Rules for creating this state machine in the preferred embodiment are: any activity in the gap indicated by the ranging status message that does not indicate the RU's temporary ID indicates a collision; a simple binary stack contention resolution algorithm is used where once an RU starts ranging, any subsequent collision push it deeper on the stack and any empty gap pops it closer to the top of the stack as in a LIFO mechanism. The ranging state machine 510 receives as its input on bus 512 from

CPU 405 a P parameter which sets the power of the ranging pulse and data which defines the barker code of the ranging pulse. Circuit 510 also receives on bus 512 RU/CU information which tells the circuit 510 whether it is in an RU or CU. The data on line 512 also controls whether a single barker code is transmitted or a specific sequence of barker codes during successive gaps to make up the authentication or signature sequence. The data on bus 512 also controls the position of a barker code pulse relative to the center of the gap. Since this data comes from the CPU 405, the CPU knows when the transmitter is ranging and can properly interpret ranging status messages broadcast by the CU and received by the CPU via bus 1096 and command, communication and control circuit 860 in Figure 12. Circuit 510 carries out the ranging process including contention resolution, pulse position modulation, steering and signature transmission described elsewhere herein in some embodiments, and in other embodiments, these processes are carried out by the CPU 405 and circuit 510 in cooperation with each other.

In some embodiments, circuit 510 in Figure 11 also plays a role in the upstream equalization process. Upstream equalization is the process of reducing or diminishing undesired noise in the desired upstream data caused by, for example, reflections from impedance discontinuities in the coax or other media, misalignment of frames etc. Equalization is implemented in part by circuit 510 in placing a particular, predetermined pattern of signals in one or more gaps between frames so that the CU and RU receivers can determine the noise characteristics then present in the channel and take steps to "equalize" or reduce the noise. In some embodiments, this is done by the RU adjusting coefficients of an adaptive filter so that it has a transfer function which is the inverse of the transfer function of the channel, i.e., the transfer function of the equivalent circuit representing the media connecting each RU to the CU. Performing equalization increase the overall system throughput capacity, but it is not absolutely essential to practice the invention if lower capacity can be tolerated. Likewise, the ranging process can be eliminated, but this also reduces the payload carrying capacity of the system.

Block 514 on the left side of Figure 11 is a register or memory storing command and control data such as the pilot channel signal to be transmitted on the 16 access and command and control channels. This data arrives on bus 399 the CPU 405. Block 516 is a multiplexer which selects between the payload data for the 128 payload channels from the framer 508 on bus 518 or command and control data on bus 520. The selected data stream is then output on bus 522. Typical command and control data includes data messages exchanged between the RU and CU regarding ranging such as "I want to start

ranging", "I found more than one barker code in the gap, please perform your contention resolution procedure" etc.

Bus 522 is coupled to a randomizer machine 524. The purpose of the randomizer is to pseudorandomly scramble the incoming data so as to make it look more like white noise. This reduces the dynamic range at the output of the transmitter. The randomizer receives its scrambling instructions from a scramble register 525 which receives and stores a seed code on bus 529. In some embodiments, the randomizer 524 can be omitted.

Convolutional Trellis encoder 526 serves to receive the stream of tribits on bus 509 and add a redundant 4th bit to each in normal operation mode. Because the 4th bit to be added to each tribit depends upon the state of the tribit from this channel during the last symbol, a memory 528 is used to keep a record of the state of each channel's 4 bit chip state during the last symbol transmission. This information is supplied to the convolutional encoder via bus 530 as each channel's tribit is encoded during each symbol. The encoder has three modes previously described, and the diversity shuffler 506 controls the mode by a signal on bus 534.

Media Access Control

Block 540 represents circuitry to acquire an access channel and carry out media access control communications to implement ISO MAC layer protocols. Since there are only 4 access channels across which all message traffic requesting channel bandwidth and awarding same, contentions will occur when more than one RU simultaneously requests bandwidth on the same access channel. Therefore, access channels are acquired according to the following protocol. Each RU transmitter receives a seed number on bus 550 and pseudorandomly selects which access channel to attempt to use and pseudorandomly selects which 6 symbols of a superframe comprised of 12 symbols to send. The RU then sends an authentication code identifying itself in the form of the unique sequence of 6 of the 12 symbols of a superframe of 4 frames, said unique sequence pseudorandomly selected using the seed. All RUs use the same seed, so the likelihood of more than one picking the same authentication code is small. The 6 symbols sent can contain the RU's message telling the CU how many channels it needs, or a separate message can be sent after access is achieved. The CU listens on all access channels, and during each superframe determines if more than 6 symbols were sent. If so, the CU broadcasts a message on the control channel indicating there is a contention on a particular access channel. The RUs trying to gain access then do the contention resolution protocol described elsewhere herein used for ranging. If only 6 symbols are detected during the superframe, the CU broadcasts a message on the control channel indicating which 6

symbols were found. The CU can include in the broadcast message code assignments for the requested channels in reservation embodiments or, in another embodiment, can simply transmit updates to the timeslot activity table indicating which timeslots or channels have been awarded to the RU which gained access. The RU that sent these six symbols then knows that it has been awarded access, and updates its timeslot activity table which is maintained in the diversity shuffler 506. All RUs hear the timeslot activity update message and similarly update their timeslot activity tables.

Once an access channel is acquired, circuit 540 may, in some embodiments, present data on buses 542r and 542i to multiplexer 544 which comprise access control messages that are sent on the 4 access channels. Multiplexer 544 either selects these media access messages on buses 542r and 542i or the encoded chips from the convolutional Trellis encoder 526 for presentation to the code division multiplexer 527 via buses 546r and 546i and buffer 548. The multiplexer 544 is controlled by switching control signals from the CPU 405 to edit the contents of the buffer 548 to overlay the 4-bit groups of the access control symbols with the payload data on bus 507 so that the media access control 4-bit groups go into the correct addresses of the buffer 548 so that they are encoded by the CDMA codes assigned to the access channels.

Because a reservation scheme is implemented in the preferred embodiment, no contentions occur on the 140 non media access control payload channels so no contention resolution protocols are carried out for these channels since there will be no contentions.

Spreading of the chips from the convolutional encoder is done by orthogonal code multiplexer 527. This circuit or software routine performs code division multiplexing or orthogonal encoding of the data on each channel by matrix multiplication. It sets the amplitude of the output chips on buses 558r and 558i based upon matrix multiplication of the orthogonal codes times the elements of the input information vectors on buses 549r and 549i from buffer 548. Each of the information vectors on buses 549r and 549i is individually spread by the orthogonal code multiplexer to generate individual inphase and quadrature result vectors.

There is only one orthogonal, cyclic code that has 144 different codes. That code is used and is, in hexadecimal representation: 0218 A503 BA4E 889F 1D92 C1F3 AB29 8DF6 ADEF. Although cyclic codes are used in the preferred embodiment for ease of implementation, any other orthogonal, noncyclic code set can also be used in alternative embodiments, or other orthogonal, cyclic codes can be used where fewer channels/timeslots are required. The cyclic code given above uses the convention that all logic 0's represent -1s and all logic 1s represent +1 in the orthogonal code spreading

matrix. The first code of the 144 different codes in the code set will be all 1s regardless of the contents of the code given above. The second code in the code set is the code given above: 0218 A503 BA4E 889F 1D92 C1F3 AB29 8DF6 ADEF. The third code is obtained by shifting the code one binary place and taking the overflow bit that "falls off" the most significant bit position edge of the code in the second least significant bit position.

The results of the matrix multiplication performed in the orthogonal code multiplexer 527 are coupled via buses 558r and 558i to one input of a switching circuit 556 switching of which is controlled by the CPU 405. The other input of the switching circuit 556 is coupled to buses 558i and 558r to receive the ranging data from ranging circuit 510. The switch 556 selects the data on buses 558r and 558i for coupling via buses 557r and 557i, respectively, to a precode FFE/DFE filter 563 during the three symbol transmission times of each frame when payload data is being sent. The switch 556 selects the ranging pulse data on bus 560 during the gap following transmission of the last symbol in each frame.

Equalization, as that term is used herein, is the process of compensating for distortions and noise that occur caused by noise in the channel between each RU and the CU. The precode filter 563 performs a measured predistortion at each RU transmitter so that the data arrives at the CU undistorted despite the channel impairments between that particular RU and the CU. The amount of the predistortion is calculated by each RU to substantially or exactly compensate for the current distortion conditions existing in the channel between it and the CU. The predistortion characteristic is implemented by setting the transfer function of the precode equalization filter 563. This transfer function is controlled by the RU/CU Coefficient data input to the filter on bus 561. Each RU uses its own unique, measured RU/CU Coefficient data to establish a predistortion which is appropriate to its own signals for its position on the network so as to cause its signal to reach the CU with little or no distortion.

The output of the precode filter on buses 562r and 562i is applied to a scaler amplifier 564 which scales the amplitude level of the digital numbers on buses 562r and 562i in accordance with a signal on bus 566 which indicates the activity level of the modem, i.e., how many timeslots are currently in use by this modem. The purpose of this scaling is to enhance performance by taking advantage of the full precision of a digital to analog converter 576 at the output of the transmitter.

The output of the scaling circuit on buses 568r and 568i are coupled to shaping filter 570 which doubles to perform carrierless amplitude and phase modulation. There are two filters in the shaping filter which have transfer functions which are the Hilbert transform of each other and which have rolloff characteristics set to digitally filter the

data on buses 568r and 568i to limit the bandwidth of the signal on each bus to the width and center frequency of the 6 MHz channel devoted to digital data communication on the coaxial cable or other media 24. The shaping filter has a squared raised cosine filter characteristic suitable to shape the outgoing chip pulses so as to satisfy Nyquist criteria in a known manner so as to provide optimal signal-to-noise enhancement and so as to minimize intersymbol interference. The filters in shaping filter/modulator 570 can have other transfer functions also which shape the chips to be transmitted such that the spectrum of the outgoing signals satisfy the Nyquist criteria.

The output of the filter/modulator is coupled on bus 574 (the filter/modulator 570 sums the orthogonal real and imaginary signals after filtering to generate a single signal on bus 574) is coupled to the input of the digital to analog converter 576 for conversion to an analog signal for application to the input of an up/down frequency converter 577. The purpose of the up/down frequency converter is to convert the frequency of the transmitted signal to the frequency allocated for upstream or downstream transmissions as the case may be in accordance with the frequency plan for the shared transmission media. The up/down converter outputs its signal on the transmission media 24 such as coaxial cable, cellular system, satellite uplink etc.

Preferred RU Receiver Block Diagram

Referring to Figure 12, there is shown a block diagram of the preferred organization for a receiver for the RU and CU modems. The quadrature amplitude modulated combined carrier arrives at the receiver on coaxial cable 24 or other media. An RF synchronous demodulator section 750 synchronously detects the QAM modulation using a local oscillator signal on line 762 which is synchronized in phase and frequency to the pilot tone from the CU in the case of an RU receiver and which is synchronized to the preamble data sent in each timeslot in the case of the CU by frequency synthesizer 760. The RF demodulator 750 outputs an analog signal on line 752 carrying the chip amplitude information for all time slots. The RF demodulator section 750 also includes a passband filter having a center frequency centered on the frequency of the 6 MHz wide band carrying the chip data and having a 6 MHz bandwidth. The RF demodulator section also includes a variable gain amplifier that has a gain control input coupled to line 758 coupled to automatic gain control circuit 756.

The signal on line 752 is converted to digital information by A/D converter 754 which performs IF sampling as is known in the prior art which was first described by Colinberg, whose papers are hereby incorporated by reference. The sampling rate is 4 times the symbol period. The advantage of using IF sampling is that it allows the use of one A/D converter to sample both the sine and cosine carriers. In alternative

embodiments, two A/D converters may be used, each having a sample rate substantially greater than the symbol period.

Phase separation of the sine and cosine components of the QAM modulated data represented digitally on bus 760 is performed by matched filter 761. The matched filter has two filters which have filter characteristics that are the mirror image of the squared raised cosine filter characteristics of the filters 1134 and 1136 in the shaping filter/modulator 570 shown in Figure 17. The matched filters separate the orthogonal real and imaginary components in the received signals and transmit them to the frame detector via buses 906 and 908 in Figures 12 and 13. The filter characteristic of the matched filter is established by data from the CPU 405 on bus 1080. In the preferred embodiment, the output of the matched filter 762 on bus 840 is filtered by an FFE/DFE filter 764 which functions to cut down on precursor and postcursor intersymbol interference. The FFE/DFE filter 764 is comprised of an FFE and DFE equalizer, and each of the FFE and DFE equalizers is an adaptive FIR filter. Adaptive FIR filters and many of the other digital signal processing components of the circuitry disclosed herein are known and are discussed in detail in Elliott, *Handbook of Digital Signal Processing: Engineering Applications*, (Academic Press, Inc. San Diego, 1987), ISBN 0-12-237075-9, which is hereby incorporated by reference. In the preferred embodiment, the FFE filter 764 is placed between circuits 765 and 767 to filter the data on bus 769.

Next, despreading of the data and reassembly of the appropriate data into the corresponding timeslots to undo the code shuffling that happened in the transmitters is performed. The first step in this process is accomplished by CDMA DEMUX (demultiplexer) 766. This multiplexer multiplies the incoming data by the transpose code matrix C^T of the code matrix used by CDMA MUX 527 in the transmitters represented by Figure 11. The resulting despread data is stored in buffer memory 768 sequentially in the order of the individual code multiplications. The CDMA DEMUX 766 or control logic 1082 generates suitable read/write control signals to cause buffer 768 to sequentially store the despread data on bus 776 output by the CDMA DEMUX 766. A deshuffler circuit 770 receives the same seed number on bus 772 as was received by code diversity shufflers 506 in the transmitters. The seed number is sent on the control channel, and is relayed to circuit 770 by the CPU 405 (not shown). The deshuffler uses the seed number to generate the same pseudorandom numbers as were generated from this seed during every symbol time by the transmitter. These pseudorandom numbers are used to generate read address pointers on address bus 774 which are coupled to the address port of buffer 768 along with suitable read/write control signals. The data stored at the addresses indicated by the read pointers is then output by the buffer on bus

795. This bus is coupled to one of two inputs of a switch/multiplexer 791. Because the address pointers are generated in the same sequence as in the transmitters when shuffling data, the data read out of the buffer 768 is read out in the correct sequence to put the despread data back into the sequential order of the timeslots.

Other data received by the code shuffling circuit 770 on bus 772 are the Tss data indicating which timeslots are assigned to the RU, and RI indicating which codes are reserved and cannot be used by this RU or CU.

This deshuffling operation is not necessary if the receiver is located in an RU because the CU does not use code hopping for data it sends to the RUs. Therefore, in some embodiments of RU receivers, buffer 768 and deshuffler 770 do not exist. In other embodiments, they do exist, but are not used and a switch 791 guides the despread data on bus 776 from the CDMA DEMUX 766 around buffer 768 and directly into the input of the amplifier 788. An RU/CU signal on line 793 controls the state of switch 791 such that either the data output bus 795 of buffer 768 or the bus 776 is coupled to input 789 of the amplifier 788. If the receiver is in a CU, bus 795 is coupled to bus 789, while if the receiver is in an RU, bus 776 is coupled to bus 789.

In some embodiments, the despread data on bus 776 is simultaneously read by a crosstalk detector (not shown) which functions to determine the amount of interference between adjacent codes and also plays a role in clock recovery so that all RU and CU receivers and transmitters can be synchronized to the same clock. Crosstalk between channels encoded with adjacent cyclic, orthogonal codes always comes from adjacent channels and happens when the data encoded with adjacent cyclic CDMA codes do not arrive precisely aligned in time. In other words, to have zero crosstalk, the clock time at which the first chip of a symbol transmitted on one channel spread with a cyclic CDMA code arrives at the receiver must be exactly the same time as the clock time at which the first chip of a symbol transmitted on an adjacent channel spread with an adjacent cyclic code arrives. A slippage of one chip clock means complete overlap and total crosstalk since adjacent cyclic codes are generated by shifting the code by one place to the right. A slippage or misalignment of less than one complete chip clock will mean that some crosstalk exists. The crosstalk detector of these alternative embodiments detects the amount of crosstalk affecting each chip of each channel by subtracting the amplitude of the chip of the channel currently being processed from the amplitude of the corresponding chip encoded on the immediately preceding channel.

In these alternative embodiments, the amount of crosstalk is sent as a clock tracking error to a control loop logic 781 which outputs a clock phase/frequency correction voltage on line 782. This signal is coupled to the phase/frequency control

input of a voltage controlled crystal oscillator 784 which generates a chip clock reference signal on line 786. This chip clock reference signal is fed to one input of a switch 787, the other input of which is coupled to receive an external clock reference signal at 8.192 MHz. A switching control signal on line 791 from the CPU 405 controls whether switch 787 selects which of the chip clock reference signals on lines 786 or 789 for output on bus 793 to phase lock loop (PLL) 880. This PLL 880 multiplies the clock reference signal on line 793 to generate two output signals at 114.688 MHz and 57.344 MHz which are supplied on bus 888 to a time base generator 886. The time base generator generates the various clock signals needed for synchronization of the system.

In the preferred embodiment however, clock recovery is performed in the RUs by frame detector 882 using the fine tuning circuitry shown in Figure 13. This circuitry generates a clock steering tracking error signal on line 900 in Figure 12. This clock steering signal is input to the digital equivalent of an integrator in control loop 781 which serves as a loop filter for a phase lock loop including vcxo 784. The averaging process of integration eliminates the random noise. The integrated error signal is output as a clock phase steering signal on line 782 to the error signal input of vcxo 784 to generate the clock reference signal on line 786.

Although a global automatic gain control adjustment was made by AGC 756, data is being received from many different RUs located at many different positions on the network. To minimize errors in interpretation of the upstream received data caused by amplitude variance caused by differing path length losses from the various RUs and channel impairments, a separate gain control adjustment is desirable for each RU. This is done by transmitting from each RU a preamble of known data before the payload data for each timeslot assigned to that particular RU as mentioned above. Therefore, a variable gain amplifier 788 is employed to amplify each timeslot's data individually. The control loop logic 781 assists in this process by sending a desired gain signal on line 790 to amplifier 788 based upon inputs received on buses 792 and 794. The input on bus 792 is data identifying which particular timeslot's data is currently at the input 789 of the amplifier 788 and is generated by deshuffling circuit 770. The control loop 781 also receives an input from control logic 1082 and CPU 405 which indicates when preamble data for a particular timeslot is being received. The input on bus 794 is generated by a memory 796 which stores individual gain control and phase error correction numbers for each of the 128 payload channels (or all 144 channels in some embodiments).

During reception of preamble data, the control loop 781 cooperates with the slicer 800, the G2 amplifier 788 and the rotational amplifier 765 to carry out an iterative process to reduce the slicer error to as low a value as possible by adjusting the amplitude error and phase error coefficients in equation (5) given above. Specifically, the CPU 405 and control logic 1082 will signal the control loop 781 and slicer 800 when preamble data is being received by an input which is not shown. Notification to the slicer 800 in Figure 12 and slicer/detector 466 in Figure 3 takes the form of activation of the CU Preamble signal on line 1086. When preamble data is being received, the control loop will set initial values for the 1/a and $e^{-j\phi}$ amplitude and phase error correction factors of equation (5) and transmit these on buses 790 and 802, respectively, to the G2 amplifier 788 and rotational amplifier 765. These circuits will operate on the received data samples to make amplitude and phase error corrections, and the slicer will compare the received signal to the 3-j constellation point it knows it is supposed to be receiving during the preamble. The amplitude and phase errors between the actual received data and the 3-j point are output on bus 798 to the control loop 781. The control loop 781 examines these error values, and adjusts the 1/a and $e^{-j\phi}$ amplitude and phase error correction factors in an appropriate direction to tend to minimize the slicer error. The process repeats itself for the next preamble 3-j constellation point. Eventually, the control loop finds values for the 1/a and $e^{-j\phi}$ amplitude and phase error correction factors that minimize the amplitude and phase error values on bus 798. These values are then recorded in memory 796 in Figures 12 and 3 as the 1/a and $e^{-j\phi}$ amplitude and phase error correction factors to use in receiving data for the timeslot(s) assigned to this particular RU. The process is repeated each time the RU is reassigned to a new timeslot(s).

The process described above regarding synchronization in the upstream to the preamble data gives upstream carrier recovery synchronization. Frame synchronization and chip clock synchronization are done in the CU for the upstream data by the frame detector 882 using the coarse and fine tuning circuitry of Figure 13. The CU receiver knows when the gap is, so the frame detector 882 in the CU does chip clock synchronization and looks for ranging barker codes and supports the process of instructing the RUs on how to alter their transmit frame timing delay values T_d so that their barker codes hit the gap.

After synchronization to the preamble in the upstream data, the CU receiver control loop 781 uses the information received on bus 792 regarding which timeslot's data is currently being received to generate an address pointer to that timeslot's

amplitude (1/a) and phase error ($e^{-j\phi}$) correction coefficients in memory 796. The control loop 781 then sends the address pointer to memory 796 via bidirectional bus 794 along with suitable read/write control signals and receives from the memory the amplitude and phase error correction coefficients for the particular timeslot being received. The control loop then places the amplitude and phase error correction coefficients on buses 790 and 802, respectively, to control the digital amplification process carried out by the amplifier 788 and the phase error correction process carried out by the rotational amplifier 765.

When the receiver is located in an RU, the multiple timeslots being received from the CU all originate from the same location and the same transmitter so they all need to be amplified by the same gain. Therefore, in an RU receiver, memory 796 need only be a register that stores one amplitude and phase error correction value used by that particular RU to receive CU data in all timeslots.

The slicer 800 is of conventional design, and includes circuitry to measure both gain and phase error for each channel's data. These errors are measured by circuitry in the slicer which compares the amplitude and phase of a received chip to the amplitude and phase of the legitimate constellation point in the constellation of Figure 5 which the received chip is supposed to represent. Recall that the constellation of Figure 5 represents all the permissible 4 bit chips that can be part of a symbol. Each chip is comprised of 2 bits plus a sign bit which define the real or I axis coordinate and 2 bits plus a sign bit which define the imaginary or quadrature Q axis component. Therefore, in polar coordinates, each constellation point has an amplitude and phase the combination of which defines the constellation point. For example, in Figure 5, chip 0010 has a magnitude and phase represented by vector 801. Assume that chip 0011, after transmission losses, crosstalk etc. get demodulated and the I and Q components after demodulation map to point 803 in the constellation having a magnitude and phase represented by vector 805. The circuitry in slicer 800 responsible for quantifying the magnitude and phase errors compares the magnitude and phase of vector 805 to the magnitude and phase of vector 801 and generates amplitude error and phase error signals on bus 798 from the differences.

The phase rotation circuit adjusts the amplified data on bus 789 representing each received chip so as to rotate the phase thereof to correct the phase error for that received chip. This is done by a matrix multiplication of the complex number representing each chip by cosine (ø) + j sine (ø) where ø is the amount of desired phase correction.

The control loop 781 also uses the phase error data on bus 798 to generate a local oscillator steering voltage on line 806 to alter the phase and/or frequency of a 3.584 MHz reference clock output generated on line 810 by a voltage controlled crystal oscillator 808 (vcxo). The clock steering signal on line 806 is a carrier tracking error derived from the pilot channel signal. The pilot channel signal carries the time synchronization sequence mapped onto a qpsk constellation. The carrier tracking error is extracted based upon a decision directed discriminator. Carrier recovery is started immediately after the AGC gain is set and ranging has achieved frame synchronization. The carrier recovery circuitry just described is monitored by the modem software to insure that it remains in synchronization, and if lock is lost, an interrupt occurs which causes re-initialization of the modem to be started and the modem transmitter to be disabled. The same is true if clock synchronization is lost, i.e., the RU local clock is locked to the CU clock and the clock recovery circuitry is monitored to make sure clock synch is not lost.

Once carrier recovery has been achieved, the kiloframe data encoded in the pilot channel is recovered to achieve kiloframe synchronization so that the RU modem registers and software can be initialized to beginning counting CU frames so as to be able to keep straight which assigned codes from CU messages are to be used during which frames. The RU receiver decodes the synchronization sequence data on the pilot channel using a bpsk constellation.

Returning to the discussion of Figure 12, the carrier reference frequency on line 810 is used by frequency synthesizer 760 to generate local sine and cosine carrier signals on line 762 that match the frequency and phase of the local oscillator carrier signals used in the QAM modulators in the transmitters (corresponding to carrier on line 427 in Figure 8). The control loop 781, vcxo 808 and the frequency synthesizer 760 combine in the embodiment of Figure 12 to perform the function of the carrier recovery circuit 515 in Figure 3.

The receiver of Figure 12 uses two feed forward equalizers (FFE) and two decision feedback equalizers (DFE). The first FFE and DFE are shown combined as circuit 764 just after the matched filter 761 and just before the orthogonal code demultiplexer. The second FFE is combined with a rotational amplifier in circuit 765 after the orthogonal code demultiplexing operation and before the slicer. The second DFE is circuit 820. The equalization process involves some interplay between these FFEs and DFE. Both of the FFEs function to eliminate or substantially reduce precursor intersymbol interference, and both DFEs function to reduce or eliminate post cursor intersymbol interference.

The DFE circuit 820 receives as one of its inputs the decision data output by slicer 800 on bus 836 and processes these signals in accordance with the filter transfer function established by the tap weight coefficients received on bus 842 from a least means square calculation circuit. The resulting signals are output on bus 846 to the subtraction input of difference calculation circuit 767. The DFE and difference calculation circuit combine to subtract out that portion of the intersymbol interference produced by previously detected symbols from the estimates of future samples.

All the DFE and FFE circuits are FIR filters with adaptive tap coefficients. The DFE circuit 820 and the FFE circuit 765 (circuit 765 is an FFE only during the equalization training period and is a rotational amplifier during payload data reception after training) receive their adaptive tap coefficients on buses 842 and 838, respectively, from the least mean square calculation circuit 830. The FFE/DFE circuit 764 receives its tap coefficients via bus 844 from the least mean square calculation circuit 830. The FFE and DFE FIR filters are given initial values for their adaptive tap coefficients that are close enough to allow the adaptation process to proceed. These preset coefficients are supplied from the CPU 405 via buses 824, 821 and 822. Thereafter, the coefficients are adaptively altered by signals on buses 842, 838 and 844 by the least mean squared circuit 830 using a conventional precursor and post cursor ISI elimination tap coefficient calculation algorithm.

The least mean square (LMS) circuit 830 iteratively calculates the new tap coefficients in a conventional manner and interacts with the FFEs and DFEs in the manner described below in the equalization section. The LMS starts with the initial tap weights and iteratively calculates the convolution sum between the tap input signals (input signals to each stage of the tapped delay lines) within the FFE 765 and the DFE 820 and the tap coefficients of the FFE 765 and DFE 820, all of which are obtained via bidirectional buses 842 and 838. The LMS then receives error signals on bus 831 calculated by difference calculation circuit 832 defined as the differences between the desired data points on bus 836 and the received data points on bus 834. The LMS then calculates new tap weights by multiplying the error signals times the corresponding tap input signals used to calculate the convolution sum times a predetermined step size which sets the rate of convergence to a stable value, and the result is added to the old tap weights to arrive at the new tap weights. These new tap weights are then sent to the FFE 765 and DFE 820 for use during the next iteration.

The LMS circuit implements a calculation which is based upon the fact that the needed change in the adaptive coefficients to the adaptive FIR filters 764 and 820 is proportional to the error on bus 831 times the conjugate of the data being input to the

filters. In other words, the error is multiplied by complex numbers representing the received chips which have had the signs of their Q or imaginary components inverted.

The DFE filter eliminates or reduces post cursor interference by supplying a subtraction value on bus 846 to subtractor 767. The data sent by the DFE filter on bus 846 is subtracted from the data on bus 769 output by the FFE filter 765 during the equalization training interval. Eliminating the precursor interference and post cursor interference from the data on the bus 834 allows the slicer 800 and a Viterbi Decoder 850 to make better decisions about what chips were actually sent despite the channel impairments. The LMS, DFE and FFE circuits can be eliminated in some simple embodiments with, for example, only 4 points in their constellations. But to get more data throughput, more complex constellations are needed, and in such a situation, the points are closer together and ISI interference makes decisional discrimination between the constellation points more difficult. This creates a need for the above described ISI elimination circuitry.

After correction for ISI interference, the corrected data is passed via bus 834 to slicer 800. The purpose of the slicer is to make instantaneous decisions regarding which point in the constellation each chip represents for purposes of generating the gain and phase errors needed by the control loop and for purposes of generating the desired data signals on bus 836. The slicer does not make use of the 4th redundant bit in each chip for this purpose, and, as a result, makes errors in interpreting chips. It is up to the Viterbi Decoder 850 to correct these errors of interpretation.

Viterbi Decoders are well known in the art, and any Viterbi decoder algorithm will suffice for purposes of practicing the invention. Basically, Viterbi Decoder 850 and memory 852 keep track of the present and last state for each timeslot for purposes of tracing a path through a three dimensional space defined by the constellation of permissible input points stretched out over a third axis representing time which is orthogonal to the I and Q axes. There is one of these three dimensional spaces for each timeslot. By making use of the redundant bit or bits in each chip, and examining the path the states of each timeslot take through the appropriate 3-D space over time, the Viterbi Decoder makes a better informed decision as to which legitimate point in the constellation of permissible points each received code represents. The information on bus 792 to the Viterbi Decoder from the deshuffler tells the Viterbi Decoder which timeslot during which each code received on bus 836 was transmitted. The Viterbi Decoder uses this information to generate an address pointer to memory 852 pointing to the state information for that timeslot. This allows memory 852 to output the state information which is used by the Viterbi Decoder to make its analysis.

The particular trellis code selected for implementation in the invention is rotational invariant with no parallel paths and 16 states.

After the Viterbi Decoder 850 outputs the correct data for each timeslot on bus 854, deframer 856 reassembles the data into the time division multiplexed timeslots in which these same data originally arrived at the framer circuit of the transmitter for encoding and CDMA spreading. The deframer 856 also descrambles the data to undo the effects of the scrambling carried out by the scrambler 524. The resulting TDMA stream of 9-bit bytes is output on serial data format bus 858. Each 9-bit byte in this data stream is comprised of the deshuffled, descrambled three tribits into which it was originally broken in the framer of the transmitter to form the three symbols of the frame during which this 9-bit byte was transmitted.

The output bus 854 from the Viterbi Decoder 854 is also coupled to a command and control channel circuit 860 which stores and/or processes codes sent on the command and control channels in the downstream data. Some switching or multiplexing function to select the command and control codes out of the stream of data on bus 854 is provided but is not shown. Codes sent on the access channel in the upsteam or downstream data are stored and/or processed by an access channel circuit 862 which receives these codes from the output of the Viterbi Decoder 850 via bus 854. The command control code data is input to C3 circuit 860 from the Viterbi Decoder via bus 854. The CPU 405 accesses the command and control data and access channel communications from the C3 circuit 860 and the access channel circuit 862 via bus 1096. The processing of the command and control channel codes and access channel codes may also occur in circuits 860 and 862, respectively, in alternative embodiments without interaction with the CPU, or the codes may simply be buffered in circuits 860 and 862 until they can be read by a management and control process performed in the CPU 405.

The ranging process in its various embodiments described earlier herein is aided by the R/Tng circuit 763. This circuit receives an RU/CU signal on line 759 from the CPU 405 which tells the circuit whether it is performing its function in an RU or a CU. In the preferred embodiment, circuit 763 is simply a DMA FIFO which stores status information regarding positioning of the barker codes in the guardbands during the ranging and initial frame synchronization process. This status information is received from the frame detector 882 via bus 883. This data is relayed to the CPU 405 via DMA transfers over bus 755 to a memory (not shown) coupled to the CPU 405. If it is performing its function in an RU, circuit 763 stores status data generated by the frame detector circuitry in implementing any of the functions indicated for any selected one of

the embodiments of the RU in the ranging, contention resolution and authentication flow charts of Figures 2A-2C. This data may include data as to how many ranging pulses appeared in the gap and data to be sent to the ranging circuit 510 in the transmitter via bus 757 for purposes of setting transmit frame timing delay. These messages to the transmitter on bus 757 include data telling the transmitter ranging circuit 510 when the barker code or other signal from the CU has been received in each frame thereby establishing the receive frame timing reference, whether to transmit another ranging pulse after contention resolution, and how to adjust the delay factor that establishes the transmit frame timing reference before sending each ranging pulse or barker code, and, in some embodiments, what barker code to transmit.

In the preferred embodiment, command, communication and control (C3) circuit 860 receives message traffic involved in the ranging, authentication and media access control processes as detailed in the ranging flow charts and transmits this data to CPU 405 via bus 1096. Such data includes data from the CU indicating when authentication is desired and data regarding when to start sending that particular RUs authentication code. Circuit 860 also receives the authentication code broadcast by the CU after an authentication interval to determine if it is the RU that hit the gap. If so, circuit 860 sends a message to the transmitter via CPU 405 to freeze its current value for the transmit frame timing reference delay at the value last used for transmission of the ranging pulses in the authentication code sequence. The circuit 763 also monitors the control channel for instructions from the CU on how to adjust its transmit frame timing reference delay to exactly center the ranging pulse in the center of the gap.

If the signal on line 759 indicates the receiver of Figure 12 is operating in a CU, the circuit 763 carries out those functions indicated for. Circuit 763 stores data received on bus 883 regarding how many barker codes have appeared in the gap during ranging and authentication and data regarding how many RUs have hit the gap, data determining the position of the barker code(s) in the gap, and data ordering changes of position of the barker code in the gap, data resulting from scanning the gap for additional unwanted pulses at the edges of the gap. This data is read by the CPU and used to compose messages for transmission by the transmitter on the control channel such as "no codes in gap-adjust your delays and try again", "one code in gap", "multiple codes in gap-enter contention resolution", "move barker codes x chips left or right", "saw sequence xxxxxxx in gaps during authentication frames", "no activity in gap during authentication interval-reexecute your contention resolution protocols" etc.

Referring to Figure 17, there is shown the preferred form of the modulators used in the RU and CU transmitters. In the modulator of Figure 8, multipliers are used to

multiply the incoming data times the local carrier signals. The result is two orthogonal RF signals bearing the inphase and quadrature information.

This same result can be achieved in a substantially different way by using Hilbert transform filters and carrierless amplitude and phase modulation. In the preferred form of modulator 507 shown in Figure 17, the multipliers 429 and 435 and local oscillator 425 and phase shift circuit 439 in Figure 8 are completely eliminated thereby resulting in a less expensive, less complex modulator that achieves the same result as the modulator of Figure 8. Specifically, shaping filter/modulator 507 of Figure 17 receives inphase (real) and quadrature (imaginary) digital inputs (or analog) on buses 568r and 568i. Although, buses 568r and 568i are shown in Figure 17 as originating at the results array for clarity of illustration, in the preferred transmitter of Figure 11, they actually originate from the output of the scaling circuit 564. In some embodiments, the scaling circuit 564 and the precode equalization filter 563 can be eliminated where higher error rates or less payload capacity can be tolerated.

The Fourier spectrum of the baseband, orthogonally code division multiplexed data on bus 568r is a constant amplitude spectrum of amplitude A_r on the real axis. The Fourier spectrum of the baseband, orthogonally code division multiplexed data on bus 568i is a constant amplitude spectrum of amplitude A_i on the imaginary axis. The direct sequence spread spectrum techniques employed in the transmitters according to the teachings of the invention has the effect of spreading the energy of the signals represented by the information vectors from minus infinity to plus infinity at a constant amplitude. Because any 6 MHz wide section of the spectrum can be selected with a passband filter and all the channel data therein recovered, this fact is employed to simultaneously carry out carrierless amplitude and phase modulation as well as filtering to satisfy the Nyquist criteria in shaping filter/modulator 507. To do this, two shaping filters 1134 and 1136 in modulator 507 are coupled to receive the signals on buses 568r and 568i, respectively. Filter 1134 has its filter characteristics set (programmably by CPU 405 in some embodiments) to establish a squared raised cosine passband filter characteristic 1142 in the real plane of the frequency domain. The passband filter characteristic has a bandwidth of 6 MHz and is centered on an intermediate frequency Fc which is established at a frequency which can be easily and conveniently achieved in a digital filter. The output signals of the filter are ultimately sent to digital-to-analog converter 576 in Figure 11 and from there to an up/down converter 577. The function of the up/down converter 577 is to raise the frequency to a

frequency in the middle of the band devoted to digital data communication to implement the CATV or cellular system supplemental services on the shared transmission media 24. The frequency is altered by the converter 577 to a frequency appropriate to the upstream or downstream direction in which the transmitter is sending data at a frequency which is located so as to not interfere with, for example, cable television programming also carried on the same media.

Filter 1136 also has a squared raised cosine passband filter characteristic 1144, but its filter characteristic is located in the imaginary plane of the frequency domain. The passband filter characteristic has a bandwidth of 6 MHz and is centered on an intermediate frequency Fc which is easy to attain in digital filter design. To insure orthogonality between the real and imaginary data output signals on buses 1146 and 1148, the transfer function of filter 1136 is the Hilbert transform of the transfer function of filter 1134.

When the baseband spectra for the real and imaginary signal components are passed through filters 1134 and 1136, the resulting Fourier spectra of the digital data on buses 1146 and 1148 contain all the encoded information from the real and imaginary information vectors encoded by the orthogonal code multiplexer 527. These digital signals on buses 1146 and 1148 are summed in summing circuit 1150. The result is output on bus 574 to the analog-to-digital converter 576 in Figure 11 for conversion to analog signals which are then raised in frequency by frequency converter 577.

Demodulation of these spread spectrum signals is accomplished in a known manner.

Referring to Figure 18, there is shown a block diagram of an alternative embodiment of a system employing CU and RU modems according to the genus of the invention. The system comprises a CU modem 1160 coupled by an HFC(hybrid fiber coax) or wireless transmission media such as a cellular or satellite radio transmission system 1162 to one or more RU modems 1164. The purpose of the CU modem is to provide a multiple-user and /or multiple-source simultaneous digital data communication facility over a limited bandwidth channel such as 6 megahertz to one or more remote unit modems coupled to the CU modem by a shared RF transmission media.

The CU modem transmits data in the downstream direction toward the RU modems using a transmitter 1170 that uses digital data to modulate one or more radio frequency carriers that are transmitted over the media 1162 after frequency conversion by up/down frequency converter 1174 to the proper assigned downstream channel frequency. The transmitter can use any modulation scheme which can transmit a master

clock reference and a carrier reference signal to the RU modems for clock and carrier synchronization purposes there, said clock and carrier references being transmitted either in-band or out-of-band. Data is transmitted in frames which the RU receiver detects. The RU transmitter achieves frame synchronization by the ranging processes described elsewhere herein. Examples of modulation schemes that will work for the downstream direction CU transmitter are QAM, SCDMA or DMT (digital multitone transmitter). Any of the conventional transmitters described in the books incorporated by reference herein will suffice for the CU transmitter, but an SCDMA transmitter is preferred. Non-SCDMA modulation schemes can be used in the downstream direction because the noise and interference problems are less severe than in the upstream direction.

The definition of "in-band" transmission of the clock and carrier is that one or more channels which would otherwise be used to transmit payload data are dedicated to transmitting the clock and carrier signals. The definition of "out-of-band transmission is that a separate carrier or some other subchannel/sideband etc. modulation scheme is used to transmit the clock and carrier information so that no timeslot or packets that could be used to send payload data is used to send clock and carrier information. The master clock signal is generated by master clock 1176 and the carrier reference signal which is modulated by transmitter 1170 is generated by master carrier local oscillator 1178.

The CU modem transmitter has a framing/addressing/packetization circuit 1166 which functions to receive payload data at an input 1168 and organizes said data into frames and addresses the data to the proper destination RU modem and the proper peripheral device coupled to that modem. The manner in which this is done is not critical to the invention. The upstream data is transmitted by SCDMA (synchronous code division multiple access). SCDMA is defined as transmission of frames of spread spectrum signals with data from different channels spread using orthogonal pseudorandom spreading codes, said frames being synchronously transmitted from different RUs located at diverse locations such that all frames of corresponding frame number from all RUs arrive at the CU modem at the same time for despreading and decoding by the inverse code transformation that was used in the RU transmitter to spread the spectrum of the data using the orthogonal, pseudorandom spreading codes. The CU transmitter's framing addressing circuit 1166 can have the structure and operation of the framing circuit 400 in Figure 3 if the transmitter 1170 is an SCDMA or DMT transmitter. If the transmitter 1170 is, for example, a QAM transmitter, the framing/addressing circuit 1166 organizes the data into frames and places data bound

for specific RU modems into timeslots assigned to those modems. The data in these timeslots of each frame assigned to a particular RU modem will includes header bits which tell the RU modem to which particular peripheral or other destination the data in these timeslots is addressed and may include other information such as packet delimiters which define the start and stop timeslots of each packet destined to a particular peripheral or byte counts etc. which tell the RU how many timeslots of data to collect for a complete packet destined for a particular destination coupled to that RU. Basically, the function of the framing/addressing/packetizing circuit includes organizing the payload data such that information as to which remote unit modem and peripheral each payload data byte is directed to can be determined.

The CU modem receives upstream radio frequency signals modulated with digital data by the RU modems using an SCDMA receiver 1172. The function of the SCDMA receiver is to synchronously extract the payload data from the upstream RF signals. This upstream payload data was synchronously modulated onto the upstream RF carrier by an SCDMA transmitter in the RU modem followed by a suitable modulation scheme such as QAM to use the data resulting from the code transformation spreading process to control some one or more characteristics of one or more RF carriers. The CU receiver 1172 can have the structure of the receivers of Figure 3 or 12.

The RU modem 1164 has the following structure. A receiver 1190 having a demodulator and detector compatible with the type of encoding and modulation performed in the CU transmitter is coupled to the transmission media. The function of the RU receiver is to receive downstream RF signals transmitted in frames by the CU transmitter and extract payload data transmitted by the CU and any management and control data transmitted by the CU. The RU receiver also functions to recover the master clock and to recover the carrier used by the CU transmitter. The recovered master clock signal is distributed on bus 1214 to all RU circuits that need it including the SCDMA transmitter 1210. The recovered carrier signal is distributed by receiver 1190 on bus 1216 to all circuits that need it including the SCDMA transmitter 1210. Recovery of the clock and carrier signals can be performed as described elsewhere herein or in any other conventional manner described in the references incorporated by reference herein.

The RU receiver 1190 can have the structure of the receivers described in Figures 3 and 12 as well as described alternatives and functional equivalents thereof or it can have the structure of conventional receivers described in the treatises incorporated by reference herein, so long as whatever structure it has is capable of decoding and extracting the payload and management and control data transmitted downstream by the CU transmitter. The extracted payload data is output on bus 1217 for

use by peripherals and interfaces to other networks or processes represented by block 1218.

An RU transmitter 1210 receives payload data on bus 1220 from the peripheral devices or processes and organizes that data into frames of the same size as the CU frames. The data so framed then has its Fourier spectrum spread by the transmitter over a bandwidth much larger than said data originally had, usually by orthogonal code division multiple access encoding or by performing an inverse Fourier transform operation. If code division multiple access is used, the spread spectrum data is then modulated onto one or more radio frequency carrier signals using a suitable modulation scheme such as QAM16 as described elsewhere herein. This process of organizing into frames, spreading the spectrum of each frame of data and using the spread spectrum data to modulate one or more RF carriers is done synchronously using the master clock and carrier signals recovered by receiver 1190 and output on buses 1216 and 1214. The resulting RF signals are output on line 1224 to an up/down frequency converter 1226 where the frequency is converted to the designated frequency of a frequency band, usually 6 MHz in width, dedicated for upstream traffic and are then output on line 1228 to the transmission media 1162. Therefore, frequency division multiplexing for the upstream and downstream traffic is achieved. Those skilled in the art will appreciate that the system of the invention uses a combination of time division multiplexing, frequency division multiplexing and code division multiplexing to achieve high-performance, multiple-user, multiple-source bidirectional digital data traffic in a distributed communication system.

The CU modem includes a gap monitor circuit 1192 that functions to monitor the guardband or other interval included in each frame to which the RU transmitters are trying to synchronize to determine if one or more barker codes have been received. The gap monitor circuit can have the structure shown in Figure 13 or any other structure that can determine when the unique code of an RU has been received, can determine if more than one code from an RU has been received in the gap, can detect how far away from the center of the gap the received barker code is and can provide status information on bus 1196 to a computer 1194. The information to the computer 1194 from the gap monitor includes whether a barker code has been received, if more than one has been received, and, if only one has been received, how far away from the center of the gap the received barker code is. Although a computer is preferred for circuit 1194, other circuits to perform this function such as gate arrays, state machines etc. may be used to generate the management and control data on bus 1198 which informs the RUs of information they need to achieve frame synchronization. Hereafter, circuit 1194 will

be referred to as a computer. The same is true of computer 1204 in the RU. It does not have to be a computer per se but can be any other type circuit that can fulfill the function. The computer 1194 then generates management and control message data on bus 1198 which are presented at one input of a switch 1200 the switching state of which is controlled by computer 1194 to select the data on bus 1198 during the interval for encoding and transmitting data from timeslots devoted to management and control messages. Those skilled in the art will appreciate that a switching multiplexer like MUX 1200 need not be used and any other known data transfer circuit or process, referred to in the claims as a data transfer circuit, to get data from one process to another such as shared memory etc. may be used to get the management and control data transmitted by transmitter 1170 at the proper time.

The RU receiver 1190 receives these management and control messages and passes them on bus 1202 to a computer 1204 which uses the management and control data to control the ranging process carried out by said SCDMA transmitter 1210 and for other purposes. The receiver 1190 also includes a gap monitor circuit that supports a gap acquisition process to locate the time of each CU frame gap. This gap monitor circuit listens for barker code data transmitted by the CU during every gap, usually by correlating received energy against the known barker code data pattern and sends gap acquisition data detailing the receipt of correlation pulses and the relative times of their occurrence to computer 1204 via bus 1202.

Computer 1204 or other control circuitry uses this gap acquisition data to determine the time of receipt of the barker code thereby establishing a frame boundary reference for the receiver to aid it in demodulating, decoding and deframing the received data. The computer 1204 uses the receive frame timing reference during the ranging process to establish a trial and error value for the transmit frame timing delay value T_d , and sends this transmit frame timing delay value T_d on bus 1212 to the transmitter to control the delay between the time when a frame arrives from the CU transmitter, and the time the RU transmitter 1210 sends the same frame back to the CU receiver with new data therein. During the ranging process, the value of T_d is varied experimentally during successive barker code transmissions until management and control data is received by the RU modem indicating that the barker code has been centered in the CU frame gap thereby achieving frame synchronization.

As a further operation in achieving frame synchronization, the computer 1204 also enables a ranging generator circuit 1206 via signals on a bus 1208 and passes messages to the ranging generator to control its operation. The ranging generator 1206

functions to generate and send to said SCDMA transmitter data defining a barker code for transmission during a ranging process to establish frame synchronization and the unique on-off morse code signature sequence of barker codes transmitted during a signature sequence of gaps that is used to achieve identification/authentication of each particular RU during the ranging process.

Computer 1204 also generates and sends management and control data to the RU SCDMA transmitter 1210 via bus 1212. This management and control data can include requests to start ranging, requests for more bandwidth, messages relinquishing bandwidth etc for various species within the broad genus of the invention.

The payload data extraction process is done synchronously in the CU and RU modem receivers. "Synchronously" as that word is used in the claims means the following forms of synchronization are practiced in the RU and CU SCDMA receivers and the RU SCDMA transmitter. The RU transmitter uses the recovered clock and carrier reference signals to drive its digital circuitry and modulator in synchronism with the CU master clock and master carrier. Synchronous or coherent detection is performed in the CU modem receiver using the local carrier signal on line 1180 or a recovered carrier from either an in-band source like the pilot channel described elsewhere herein or some out-of-band source. The CU's SCDMA receiver uses its own master clock and master carrier without recovering either from the signals transmitted by the RU. This provides knowledge in the CU SCDMA receiver of the RU's SCDMA transmitter carrier phase and frequency because the RU transmitter does a carrier recovery or carrier synchronization process to recover the carrier used by the CU transmitter for purposes of synchronizing the RU SCDMA transmitter. An RU SCDMA or other type of receiver recovers the master carrier reference from, for example, the pilot channel transmitted by the CU and recovers the master clock reference from the barker codes sent by the CU during the gaps of every frame. Those recovered clock and carrier signals are used to synchronize the detector in the RU receiver and are used by the RU SCDMA transmitter. Frame synchronization is also part of the synchronization implied by the term "synchronously" in the claims, and is achieved by the trial and error process of adjusting the transmit frame timing delay of the RU SCDMA transmitters as described elsewhere herein but can also be achieved with alternative ranging techniques where the CU instead of the RU does the ranging calculation and instructs the RU what transmit frame timing delay to use. In these embodiments, the RUs transmit a signal which is easily recognizable above the noise by the CU. The CU then determines the identity of the RU in any way, calculates how far off the center of the gap the RU's signal is and

instructs it how to adjust its delay to achieve frame boundary alignment of the RU frames with the CU frames.

Referring to Figure 19 there is shown a block diagram of a synchronous TDMA system for bidirectionally communicating digital data over any transmission media including hybrid fiber coax using FDMA upstream and downstream channel separation so as to not interfere with other services such as cable television programming sharing the HFC. The CU modem 1380 receives a TDMA stream of data from higher level software layers, peripherals or other interfaces such as a T1/E1 line, and synchronizes its own master clock 1384 from signals on the TDMA bus 1382 that define the frames of timeslots thereon. The TDMA stream on bus 1382 is received by a CU TDMA transmitter 1386 which also receives a master clock signal on bus 1388 and a master carrier reference signal on bus 1390 from a master carrier reference oscillator 1392. The TDMA transmitter receives the frames of data and modulates the data from each timeslot of each frame onto one or more carrier signals supplied by the master carrier oscillator 1392 using any modulation scheme which can transmit the master clock and a carrier reference signal to the RU modem either in-band or out-of-band. Examples of such modulation schemes include QAM, QPSK etc. For example, one or more time slots may be devoted to sending data encoding the master clock signal and master carrier reference. The modulated RF signals are output on line 1394 to an up/down frequency converter 1396 which converts the frequency thereof to a downstream frequency which will not interfere with other services sharing the transmission media 1398 such as cable TV programming fed into the media from bus 1400. The frequency converted signals (frequency conversion is optional if the master carrier in the CU modem can generate a carrier at the desired downstream frequency and the upstream channel can be some frequency which can be synchronized to the downstream frequency such as a harmonic) are output on line 1402. An RU modem 1404 receives the downstream data on line 1408. A TDMA receiver coupled to line 1406 recovers the master clock and master carrier reference signals using any conventional circuitry or the circuitry and methods disclosed herein and outputs the recovered clock signal on line 1410 and outputs the recovered carrier signal on line 1412. The recovered payload data is reassembled into a TDMA data stream and output on bus 1414 to peripherals or other interface processes.

Those peripherals or other interface processes also supply a TDMA input data stream on bus 1416 to an RU synchronous TDMA transmitter 1418. This transmitter receives the recovered clock and recovered carrier signals on lines 1410 and 1412, respectively, and synchronously organizes the TDMA input data on bus 1416 into TDMA frames having the same duration as said CU frames. These frames are then modulated

onto one or more carrier signals using the same or a different modulation scheme used by the CU transmitter, and the frames of modulated RF signals are transmitted to the CU in frame synchronization with the CU, i.e., the frames are transmitted from the RU transmitter with a transmit frame timing delay set for this particular RU's position in the system relative to the CU such that the frames transmitted by the RU arrived at the CU aligned with the CU frame boundaries. All RU modems in the system have their transmit frame timing delays set for their particular positions on the network so that all their frames arrive at the CU aligned with the CU frame boundaries. The modulated RF data output by RU TDMA transmitter 1418 is coupled on line 1420 to an up/down frequency converter 1422 that functions to change the frequency of the upstream channel to a frequency that is far enough removed from the downstream channel frequency and from the cable TV programming so as to not interfere therewith. The upstream data is then transmitted via line 1424 and the transmission media to a CU TDMA receiver 1426. This receiver receives a master clock signal on line 1428 from the master clock oscillator 1384 and receives the master carrier signal on line 1430 from the CU's master carrier reference oscillator. The CU TDMA receiver 1426 recovers the payload data from the modulated upstream signals and reassembles the payload data into a TDMA output data stream on bus 1432.

The TDMA transmitters and receivers in this system can be conventional, but the RU TDMA transmitter must be able to delay transmission of its frames by a variable transmit frame timing delay so that its frames arrive in frame synchronization with the frame boundaries of the CU. Any ranging process described herein or any other known ranging process can be used to achieve this frame synchronization. If any of the trial and error class of processes described herein is used, a computer 1434 in the RU modem sets an initial transmit frame timing delay either at its own initiative or upon receipt of a ranging solicitation message from the CU via a management and control data path 1436 from the receiver 1406. This initial delay value is sent to the RU transmitter via bus 1438. The CU receiver assists in the ranging process by sending data regarding what signals from the RUs it found in the frame gaps if gaps are used or what RU ranging signals were detected over the frame interval via bus 1440 to a computer 1442. The computer sends feedback ranging data to the RU via bus 1444 coupled to the CU transmitter 1386. If the class of ranging embodiments where the CU does the ranging process for the RU by determining how much the RU must move its ranging pulse to achieve frame synchronization and so instructing the RU, bus 1440 still carries data regarding what ranging pulses the CU receiver saw, but computer 1442 then figures out how much the RU needs to add to or subtract from its transmit frame timing delay and

sends a message via bus 1444 to the RU so instructing it. This message reaches computer 1434 via bus 1436, and the computer sets the instructed delay via bus 1438. Any other ranging process that can achieve frame synchronization other than the ones described herein will also suffice to practice this particular embodiment.

All of the transmitter embodiments disclosed herein can utilize an active bandwidth management process carried out by bidirectional message traffic between the remote units and central unit over the management and control channels. Remote units can request more or less bandwidth or request reserved bandwidth, and the central unit can evaluate remote unit privileges for bandwith reservation, arbitrate conflicting requests for reserved or more bandwidth and then award bandwidth in accordance with the results and send downstream management and control messages telling each remote unit which codes have been assigned to carry its traffic during which frames.

ATM PROTOCOL TRANSMISSION OVER HYBRID FIBER COAX

Referring to Figure 20, there is shown a block diagram of a preferred embodiment of a system for implementing ATM protocol transmissions over a shared hybrid fiber coax (hereafter HFC) cable television network. The circuitry to the left of the shared HFC network 1000 represents the circuitry in the CU. The circuitry to the right of the HFC media 1000 represents the Customer Premises Equipment (hereafter RU or RU) for customer #1. Although only one customer RU is shown for simplicity, in an actual system numerous RU units would be connected to the shared media 1000 simultaneously.

The CU system is comprised of a SAR circuit 1002 which is coupled to a plurality of devices and other networks that supply data for transmission to the RUs and which receive data from the RUs. For example, device #1 could be a video on demand player outputting a stream of digital data on bus 1004. Likewise, device #n could be an Ethernet or ATM network interface card coupling the SAR to a local area network (not shown) using an ATM or Ethernet transport protocol. The SAR may also be coupled to one or more service provider or other networks. For example, the block labelled network #1 may be the local and long distance telephone network interface. The block labelled network #2 may be an internet provider interface.

All of the networks and devices to which the SAR is coupled transmit and receive data byte streams bidirectionally on the buses like bus 1004 that connect the network or device to the SAR. The SAR functions to receive the data stream from each device and network interface and packetize the data from each device or network interface into 55 byte ATM cells which are output on bus 1006 in a slightly-modified, industry-standard, time-division-multiplexed Utopia + data stream. Utopia is a format for a time division

multiplexed data stream that has transmitted in its timeslots 53 byte industry standard ATM protocol cells. ATM is a network transport protocol which provides an abundance of advantages over other network protocols in delivering multimedia services. Because ATM protocols guarantee quality of service, such as guaranteed bandwidth/bit rates, services like video teleconferencing, digital video and other high bandwidth consumption services which cannot tolerate interruptions in the flow of data can only be transmitted over ATM networks. The ATM protocol provides the ability to provide integrated voice, video and data services simultaneously over a single physical media. SAR 1002 receives these digital voice, data or video bytes and packetizes them into slightly modified ATM cells having 48 bytes of payload referred to as the cell body, 5 bytes of header information which, among other things, identifies the particular device at the RU for which the payload data is destined, and 2 bytes of "virtual link" header information. The virtual link header information includes two bytes which comprise the address of a formatter at the RU which will be explained below. These two bytes indicate to which of the RUs a particular ATM cell is directed. The data from the devices or network interfaces coupled to the SAR 1002 contain destination data indicating to which RU the data is bound and to which device or network interface card coupled to the destination RU/CPE modem the data is to be directed. This destination data is used by the SAR to compose the information in the standard ATM header and the virtual link header portions of each Utopia + cell output on bus 1006 for downstream traffic (toward the RUs). This Utopia + TDMA data stream is transmitted to a multiplexer/demultiplexer (also referred to as a formatter) 1009 which implements an interface between the ATM cells on bus 1006 and the synchronous code division multiplexed physical layer described elsewhere herein.

Conversely, SAR 1002 also receives a "Utopia +" format TDMA data stream carrying upstream traffic from the RUs and depacketizes the ATM cells in the stream and uses the header information contained in the packets to transmit each cell out on the appropriate bus to the device or network interface indicated in the header information.

SARs (segmentation and reassembly circuits) are well known in the art of ATM networks, and no further details will be given here about their structure or operation.

The multiplexer/demultiplexer 1009 (hereafter sometimes referred to as the ATM-SCDMA interface 1009) receives the "Utopia +" time division multiplexed data stream from the SAR 1002 which contains all the downstream data originating in the devices connected to the SAR 1002 and destined for any RU. The function of the multiplexer/demultiplexer 1009 is to add a 9th bit to each 8-bit byte of the "Utopia +" data stream so as to encode the sequence of 9th bits in the 55 8-bit bytes of each ATM

cell so as to signal the RU where the first byte of each ATM cell starts and to add CRC or other error detection and correction code data to be used by the RU receiver in detecting and correcting errors in the header and payload information of the 55 8-bit bytes of the ATM cell. The ATM-SCDMA interface 1009 also serves to output the 55 converted 9-bit bytes of each ATM cell on bus 1011 as a time division multiplexed data stream having one 9-bit byte in each time slot in synchronization with the 8 MHz byte clock signal to which the circuitry in the SCDMA physical layer is synchronized. It is this time division multiplexed data stream of one 9-bit byte per timeslot on bus 1011 that is the raw data input to the synchronous code division multiplexing transceiver circuitry described above that implements the physical layer of the OSI model for the embodiment shown in Figure 20. All the interleaving, scrambling, code division multiplexing, code hopping, and Viterbi decoding that happens on the physical layer is effectively invisible to the Data Link and MAC layer protocols described herein. However, there is a tight coupling between the attributes and characteristics of the SCDMA physical layer and the methods of operation of the data link and MAC layer protocols of the claimed invention so as to be able to carry out an ATM protocol in the nonsymmetrical single point to multipoint downstream and multipoint to single point upstream environment characteristic of a CATV system. That coupling is described next because it is important in understanding the conceptual underpinning of the hardware and software described herein.

The ATM protocol was designed for a local area network point to point environment without a shared media. A "shared media", as that term is used herein, means a media wherein more than one pair of devices may be communicating with each other at the same time. In a typical ATM protocol local area network, all the devices are coupled together through individual connections such as twisted pair, coax, fiber optic waveguide etc. from the device to a switching hub or ATM switch. A cell of data generated by another device is sent via a point-to-point protocol in this topology by placing the destination address of the device to which the cell is addressed in the cell header and transmitting the cell to the ATM switch. The cell is then switched onto the appropriate branch connection from the switch to the device to which the cell is addressed or to a bridge or router which will ultimately direct the cell on an appropriate path to the device to which the cell is addressed. No other pair of devices can use the various data paths involved in this transaction while the transaction is occurring as that would cause a collision and loss of data. If two devices simultaneous direct ATM cells to a single device, such as two workstations directing ATM cells to a server for storage, the two ATM cells from the different work stations arrive at the ATM switch simultaneously. In such

a case, both cells cannot simultaneously be sent to the server on the single connection from the ATM switch to the server. In such a case, the ATM protocol calls for the ATM switch to buffer one cell while the other cell is being transmitted to the server. When the first cell is completely transmitted, the second cell is transmitted.

This ATM point to point data exchange protocol will not work directly in a CATV shared media environment shown in Figure 21, so it is implemented with a virtual link logical topology shown in Figure 22. In Figure 21, each branching media diverging from the CU to the RUs, such as media 1001, 1003, 1005 and 1007 is shared by the RU peripherals of all the RU coupled to each media. For example, media 1001 is shared by the peripherals connected to 4 RU, of which RU 1009 and 1011 are typical. None of these peripherals coupled to RU 1009 and 1011 could simultaneously exchange ATM cells with peripherals coupled to the CU 1013 because collisions would occur on shared media 1001 causing loss of data. These point to point communications are necessary to carry out ATM protocols however, so virtual links between each RU and the CU are established using code division multiplexing in the preferred embodiment. In alternative embodiments, the virtual links can be established using any other known form of multiplexing over the shared media 1000 that is capable of keeping the signals from the different RU's separate while allowing more than one RU to simultaneously transmit to the CU. Examples of such alternative embodiments will be given below, but generally use of TDMA systems and FDMA systems to establish the virtual links is within the genus of the invention. SCDMA is preferred to avoid the disadvantages of TDMA and FDMA systems. Each RU's virtual link to the CU is not shared and is implemented by assigning specific CDMA codes to the RU and CU transceivers. Thus, to any particular RU transceiver and the CU transceiver assigned to the same code or codes, the shared physical media looks logically like it is not shared since the CDMA encoding of the data prevents simultaneously transmitted data using other orthogonal codes from inteferring with it. Each virtual link carries the ATM compliant cells and is the sum of all data transmitted to and received from a specific RU. For purposes of discussion of the ATM implementation below, the focus will be on the logical equivalent of what is happening on the physical level, i.e., the virtual traffic on the virtual links will be described instead of the actual physical signals travelling across the shared media with interleaved and scrambled bits from every byte in all 128 timeslots in every symbol. This will make it easier to understand the ATM implementation.

In the pictorial description of the HFC network given in Figure 21, the digital backbone network is defined in the TeraComm standard HFC network topology (which is incorporated by reference herein) as providing interconnectivity between different head

ends and between each CU and various wide area networks. The Service networks provide value added services over the pure transmission technology described herein such as voice networks, video networks and pure data networks such as the Internet.

Returning to the consideration of Figure 20, the TDM stream of data on bus 1011 is received by a framer circuit in the SCDMA ASIC 1013. There it is processed and has its spectrum spread or encoded with an SCDMA orthogonal code assigned by a computer 405 to yield a result vector comprised of orthogonally encoded 4-bit chips. Each 4-bit chip is then split into two parts and used to QAM modulate two carriers which are 90 degrees out of phase. The resulting RF signals are bandwidth limited and are output on line 1017 to an RF up/down converter 1018 for translation to the frequency of the downstream signal.

Frequency division multiplexing is used to separate the upstream data from the downstream data. In the preferred embodiment, the frequencies below 54 MHz are reserved for upstream while frequencies between 54 MHz and 750 MHz are reserved for the ordinary cable television programming using NTSC video modulation and the downstream QAM modulated digital data.

At the RU, the downstream QAM modulated digital data signals are received by an RF up/down converter 1020 where they are converted in frequency to the carrier frequency used in the modulator in SCDMA ASIC 1013. The converted frequency is then transmitted on line 1022 to the SCDMA ASIC 1024 at the RU. There, a demodulator demodulates the RF in the manner described above and converts the analog signal to digital data. The digital data is then demultiplexed using the transpose code matrix, Viterbi decoded and "deframed" for reassembly on bus 1026 as a replica of the TDM data stream on bus 1011. Computer 1028 (same computer as previously described for RU receiver and transmitter sections) assists in this process by running the algorithms described above to carry out ranging, training and control operations by the ASIC to make sure the proper transpose code matrix is used for decoding in accordance with the current code assignments to the various timeslots/channels.

The TDMA stream on bus 1026 is examined by formatter ATM/SCDMA interface 1030 to determine if the virtual link address in the header indicates that the cell is addressed to the RU of which the formatter 1030 is a part. If so, the formatter reassembles the ATM cells from the bytes of data in the various timeslots of the TDMA stream on bus 1026. The logical structure of the downstream data flow is shown in Figure 23. The downstream physical layer signals are divided into frames. Each frame is composed of three symbols of 144 chips each and a gap or guardband comprised of 16 chips for a total of 448 chips each having 278 nanoseconds duration. The frame period

is 125 microseconds. The frame data payload is 128 channels times 72 kilobits per second per channel plus 16 control and management channels each of which has a data rate of 72 kilobits per second for management and control information.

On the logical level, the ATM cells are comprised of 55 9-bit byte cells which are transmitted end to end sequentially without regard to the frame boundaries except that the first ATM cell in a group of cells is transmitted synchronously with the start of whatever frame it is transmitted in. Thereafter, 2.2 cells per frame are transmitted, with the 9 bits of each byte being loaded into the framer circuitry in synchronization with the 8.192 MHz bit clock signal.

The formatter 1030 in Figure 20 is responsible for establishing the ATM cell boundaries so that they can be detected by the receiver on the other end of the HFC network. This is done with the aid of the 9th bit in each byte. Referring to Figure 24, there is shown a diagram of a typical Utopia + format 55 byte ATM compliant cell. The 8 bits of data from whatever source generated it are stored in the first 8 bits of every 9-bit byte. The 9th bit is encoded in a special way, for the downstream data, by the ATM/SCDMA interface circuit (formatter) 1009 in the CU modem 999. The formatter 1009 in the CU modem encodes the 9th bits of the first eight 9-bit bytes of the payload section of the 55 byte ATM cell with a unique sequence of bits which are defined as a start code shown at 1035. When the formatter 1030 at the RU detects this start code in the downstream data on TDM bus 1026, it knows where the ATM cell boundaries are for the payload section by counting backward 8 bytes and counting forward 55 bytes from there and also knows where the 7 header bytes start relative to the start code. The formatter then strips off the two byte virtual link header information if the ATM cell is addressed to this RU and strips off the 9th bits to leave a pure Utopia format 53 byte ATM cell. Cells reconstructed in this way by the formatter 1030 are then output in a TDM stream in pure Utopia format on bus 1033 with one 53 byte ATM cell per Utopia timeslot. From there, a SAR 1034 examines the standard ATM header to determine which device coupled to the SAR to which the cell is addressed. The cell is formatted and output to the appropriate device in the appropriate format. In the case where the destination device is coupled to the SAR by an Ethernet network, the ATM cells are packetized by the SAR into Ethernet packets and forwarded to the Ethernet interface card 1053. For illustration purposes, the SAR 1034 in the only RU shown is coupled by bus 1036 to a videophone 1038, by bus 1040 to a digital VCR 1042, by bus 1044 to an ATM local area network interface card 1046, by bus 1048 to a digital phone 1050, and by bus 1056 to an Ethernet interface card 1058. Other or alternative peripherals may also be connected.

The logical format of the upstream data from the RUs to the CU is slightly different owing to the variable bandwidth needs of the various peripherals connected to the RUs and the distribution of the available channels/codes by the CU computer 1015. The computers in the various RUs like computer 1028 monitor the state of fill of a queue buffer in the formatter 1030 to determine how backed up the buffer is. If the state of fill of the buffer gets to a certain point representing a danger of overfilling the buffer and loss of data, the computer 1028 generates a request on the shared access channels requesting more bandwidth. The computer 405 in the CU collects all the bandwidth requests, arbitrates among them based upon: the privileges of each RU requesting bandwidth to reserve bandwidth; the current bandwidth allocation scheme in use, the pending requests and the number of available channels. Awards of certain channels/timeslots are then made and messages sent via ATM cells in the downstream data channels devoted to management and control to the various RUs informing them which channels have been awarded to them. The CPU's 405 and 1028 perform the media access control algorithm in reading how much data each RU has sent and received in the last 10 milliseconds, generating and arbitrating access requests, resolving contentions on the access channels, assigning channels etc.

The channel allocation calculation is done 100 times every second (a new computation every 10 millisecond), and the new awards are distributed by a multistep protocol, each step of which takes about 1 millisecond. Therefore, the time between changing of allocations of channels can stretch out over tens of microseconds. However, actual changing of allocations occurs synchronously with a frame boundary.

Figure 25 illustrates the logical structure of the upstream data for a typical example. Suppose the RU shown in Figure 20 has been awarded channels 1, 5 and 35 in response to a bandwidth request to support digital telephone 1050. In response to this allocation via ATM cells received from computer 405 at the head end, formatter 1030 will dispatch data to computer 1028 in the RU via bus 1029. These cells tell the computer which timeslots/channels have been allocated to this RU. Computer 1028 then generates signals on bus 1031 which control the SCDMA ASIC 1024 to take data from the ATM cells in the TDMA stream on bus 1026 and distribute them in accordance with the allocation, for example as shown in Figure 25, among the various timeslots of each frame.

If only the digital telephone 1050 is in use, all the timeslots in the TDMA stream on bus 1026 will be in use to transport data from the phone. If more than one peripheral coupled to the RU is simultaneously in use, the formatter sends the cells from the various devices in use in series on bus 1026, i.e., one cell from one peripheral is

completely sent on bus 1026 before another cell from another device may be sent.

The formatter 1030 sends data to the SCDMA ASIC 1024 which informs it where each ATM cell boundary is. Signalling of the ATM cell boundaries in the upstream direction is done exactly as it was done in the downstream direction. The formatter adds 9th bits to every 8-bit byte in every ATM cell. The first 8 of these 9th bits comprise a start code, and the last few of the 9th bits are CRC data. The remaining 9th bits between the start code and the CRC data may be used for a subchannel data transmission. The SCDMA ASIC 1024 detects the start code for every ATM cell and distributes the 55 9-bit bytes of each ATM cell arriving on bus 1026 sequentially in accordance with the allocation of channels in each frame to that RU. An allocation of a channel is in effect an assignment of one or more specific orthogonal CDMA codes which may be used only by that RU and which are used by the CU receiver in despreading that RU's data only. In the case of an alternative embodiment where FDMA is used to establish the virtual links, allocation of a channel is assignment to an RU of one or more unused frequency in the upstream band of frequencies for that RU's data transmissions to the CU. In the case of an alternative embodiment where TDMA is used to establish the virtual links, assignment of a channel is the assignment to an RU of one or more unused timeslots on said shared media 1000 during which that RU may transmit its data to the CU. For an allocation of channels 1, 5 and 35, the SCDMA ASIC 1024 will take the first three 9-bit bytes of the first ATM cell and place them in logical timeslots 1, 5 and 35 of frame 0 in Figure 25. The next three bytes of the same cell will be placed in logical timeslots 1, 5 and 35 of frame 1. This process is continued until either the entire ATM cell has been transmitted in this way or the channel allocation changes. In the event the allocation changes before the entire ATM cell has been transmitted, the remaining bytes of the ATM cell are simply transmitted in the same manner using the new allocation. This process is repeated until all upstream ATM cells have been transmitted.

The ATM/SCDMA Interface 1009 at the CU extracts the bytes from a particular RU from their assigned timeslots and reassembles them into ATM cells. These ATM cells are then reassembled into a Utopia + TDMA stream and output on bus 1006 to SAR 1002 for distribution as data streams to the various devices coupled to the SAR 1002.

Figure 26 shows a diagram of how the hardware and software architecture in the CU and customer premises equipment implement the data link, media access control (MAC) and physical (PHY) layers of the OSI model. Specifically, the data link layer in the OSI model attempts to make the physical layer reliable and provides the protocols to activate, maintain and deactivate links. The principal service offered by the data link layer is error detection and control thereby allowing the next higher layer to assume

data free transmission. The data link layer protocol is implemented in the invention by the transmission of ATM cells in the manner described herein.

The MAC layer (media access control layer) is implemented by transmission of the byte stream and in band management data (the in band management data is the data of access requests, replies, contention resolution messages, channel assignment messages etc.) while the physical layer is preferably implemented using RF QAM modulation of SCDMA spread spectrum data by the synchronous code division multiplexer circuitry described generally in Figures 1-19 and implemented in the ASICs 1013 and 1024 in Figure 20. In the case of alternative embodiments where other forms of multiplexing of the shared media 1000 are used, the PHY layer of Figure 26 represents TDMA, FDMA or other types of multiplexing systems.

Figure 27 illustrates the system software architecture and the interfaces between the upper hardware and software layers in the OSI model at the CU and RU sites to the system of the invention. The upper layers of the CU equipment, represented by box 1050 representing various applications that are in execution that are sourcing data to and sinking data from the RUs, are coupled to the data link layer 1052 of the invention by a data stream formatted in Utopia +. This corresponds to the data on bus 1006 in Figure 20. The data format Utopia + on bus 1006 differs from a pure Utopia format only in that a two byte virtual link header is added to the standard 53 byte ATM cell. The data link layer 1052 is implemented by the ATM/SCDMA interface 1009. The data link layer protocols are linked to the MAC and PHY layers 1054 and 1056 by the bidirectional TDM stream on bus 1011 in Figure 20. The MAC and PHY layers are implemented by ASIC 1013 in Figure 20.

The PHY layer at the CU is coupled to the PHY layers 1058 and 1060 at two RU sites. The MAC and PHY layers at these sites are implemented by ASICs like circuit 1024 in Figure 20. The MAC and PHY layers at the RUs are linked by bidirectional TDM streams on bus 1026 in Figure 20 to formatters like the formatter ATM/SCDMA interface 1030 which implements the data link layers 1062 and 1064 at the RUs. The data link layers at the RUs are coupled to the upper layer application processes of the peripherals, represented by blocks 1066 and 1068, by pure Utopia format TDM streams of 53 byte ATM cells. These data streams are exemplified by the data on bus 1033 in Figure 20.

Figure 28 illustrates the Utopia + format ATM cell used at the CU equipment on bus 1006. The total length of the cell is 55 bytes with a two byte virtual link header 1070 having data therein which defines to which virtual link the cell belongs. The data in the virtual link header defines which RU to which the ATM cell is destined. The

number of bytes or bits in the virtual link header 1070 only needs to be enough to distinguish between the number of possible virtual links to which the cell could be assigned. In some embodiments, the virtual link header information may also identify which particular SCDMA codes are to be used in communication between the CU and that particular RU. In other embodiments, the codes to be used between the CU and each RU for any particular allocation of codes are sent separately in message traffic on management and control channels that are part of the 144 total channels. The standard ATM header 1071 contains standard ATM header information in one embodiment and identifies to which particular peripheral coupled to the RU the payload data in the main cell body 1073 is directed. Typical ATM headers include both source and destination device address fields.

Figure 29 shows the superframe structure used by the PHY layer to communicate via SCDMA over the shared HFC of the CATV plant. A superframe is comprised of 4 frames of 128 payload channels and 16 management and control channels, each frame separated from the next by a guardband. Each channel is the scrambled, interleaved equivalent of one 9 bit timeslot in the TDMA streams from the formatter or ATM/SCDMA interface circuits. The data rate is 8000 frames per second yielding a 10 megabits/second data rate, of which 9 megabits/sec is payload data. Figure 30 shows the arrangement of the 16 inband management and control channels in a single frame relative to the 128 payload channels marked D0 through D127.

The downstream data is organized as a point to multipoint communication channel providing a capability to transmit data from the CU to multiple RUs using a broadcast type mechanism. Figure 31 illustrates the PHY downstream broadcast architecture. The downstream broadcast communication channel carries only management and control information in a continuous stream of words running at 8.192 MHz on 8 of the 16 management and control channels in each frame. Each of the 8 channels carries one management and control word which is 9 bits wide, and the words are sent using a superframe of 4 grouped frames. Thus, after one superframe is sent, a 32 word management and control message has been sent to all RUs. 2000 of these 32 word management and control messages are sent every second. The virtual link header information of these 55 byte ATM cells comprising the management and control messages carry virtual link information which causes all cells to be distributed to all RUs thereby implementing a broadcast architecture. Downstream data from a peripheral or network source coupled to the CU destined for a particular peripheral coupled to a RU is sent via 55 byte ATM cell on one of the payload channels.

Figure 32 illustrates the upstream multiple access architecture implemented by

the PHY and MAC layers. The upstream communication channel is characterized by a multipoint to single point topology carried out using a combination of time division and code division multiplexing in the preferred embodiment. In alternative embodiments, SCDMA may be used alone without TDMA multiplexing of input data. For example, inputs from different peripherals coupled to an RU can arrive FDMA multiplexed or on separate inputs to the SAR which then assembles the utopia ATM cells from the different sources. In other embodiments, TDMA or FDMA may be used to establish the virtual links. Each timeslot or channel is the logical equivalent of one orthogonal CDMA code to the MAC layer. Each RU can be allocated more than one timeslot/channel/code depending upon its needs, and ATM quality of service is implemented by allowing reservation of one or more channels to one or more RUs. This provides the ability to guarantee bandwidth to RUs with high load peripherals in use that cannot tolerate interruptions in data flow. The CU MAC layer 1054 in Figure 27, by controlling code allocation, controls bandwidth allocation and quality of service.

Upstream access channels are used to carry traffic from the RUs to the CU requesting attention. Access requests can be of several types including MAC layer registration, requests for bandwidth, etc. There are 12 8-bit wide upstream access channels, only the first 6 of which are used for access requests. There is a specific access request contention resolution protocol that is described above herein that allows the access channels to be shared by all RUs and which handles contention resolution when multiple RUs simultaneously try to use the same access channel.

An RU can send an access request during any superframe. Figure 33 shows the format of an access request. The CU receives the access request and signals by a downstream message the status of each access request channel in the form: idle (for no activity on the access request channel); contention (more than one RU simultaneously requested access on the same channel) or access requested accepted (access request received and being processed). The access request is comprised of two 12 bit fields: a station ID field identifying the RU making the request; a type field indicating the type of access requested; and a checksum field which allows the CU equipment to check for contention. An access request uses 2 of the 8 available upstream management and control channels. If an RU has not yet had a station ID assigned to it, as is the case for an initial registration, it uses a random number to insure uniqueness. A collision is defined as a simultaneous access request by two or more RUs on the same channel.

When two or more RUs transmit their IDs on the same access channel at the same time, the IDs combine at the CU in such a way that the checksum fails and the CU can determine therefrom that a collision has occurred. The CU then broadcasts a code in the

downstream data that indicates a collision has occurred. The RUs resolve this contention by each waiting a randomly selected exponential delay and retrying the access requests after the delay. Figure 34 illustrates the upstream access protocol. Line 1080 represents an upstream access request launched from RU #1 toward the head end, and line 1082 represents the access request acknowledgment communication in the downstream data from the head end. Line 1084 represents simultaneous access requests from RU #1 and RU #2 causing a collision at the head end. Line 1086 represents a report from the CU in the downstream data of the collision. Each RU then waits a randomly selected delay period and then attempts the access request again, as represented by lines 1088 and 1090.

The physical layer performs a ranging and a training algorithm to achieve alignment in the time domain and power domain, respectively to improve the error rate and decrease crosstalk. These processes are symbolized by the flow chart of Figure 35 which shows the physical layer initialization sequence. First, in step 1092, each RU and the CU equipment perform initialization and diagnostics. Then, in step 1094, downstream synchronization is achieved. This step represents the process of the RUs recovering local clock synchronization from the CU clock information embedded in the downstream barker codes. This synchronizes their local clocks with the CU clock and recovers the downstream hello message which contains the upstream frequency associated with each channel. The downstream synchronization can start with a known channel, but also can scan the spectrum for a channel in an autofrequency detection scheme.

Next, as symbolized by step 1096, the ranging process described elsewhere herein is performed by the RUs. Ranging is performed fully once at power-up and, thereafter, only after complete loss of synchronization. No transmission by an RU is allowed until the ranging process starts.

The training step 1098 allows the modem to minimize the effects of line impairments by pre-compensating for them at the RU transmitter so that an optimal signal is received at the head end. Training also lets the modem set optimum power level and fine timing alignment for precise frame synchronization. Training is performed immediately after the initial ranging, and at a periodic interval while a RU has one or more active timeslots assigned to it. When the RU is idle, training is done at a slower periodic interval. A link management service is used to provide the ability to manage the network of RUs. The link management service supports basic primitives that are equivalent to the SNMP Set, Get and Trap functions. The link management service enables the transmission of management information including statistics and

configuration information for the physical and MAC layers.

MAC LAYER

The MAC layer includes all the algorithms pertaining to access to the physical layer. These algorithms include registration, authentication and bandwidth requests.

The registration algorithm is performed by RUs to establish potential usage of the physical layer. Registration involves assignment of a unique 12 bit station ID number to the RU by the head end. This process is initiated by the RU upon power up and after completion of the physical layer initialization sequence of Figure 35. The RU starts the registration process by issuing an upstream access request with a random ID (known as a temporary station ID) and a request type field (1091 in Figure 33) indicating that the access request is a registration request. The CU performs a collision detection using the access request checksum and notifies the RU upon successful completion of the collision check using a downstream message indicating access acknowledgement. Then the CU notifies the RU addressing it using the random ID of the station ID it will be assigned for the rest of the session. This is accomplished using the downstream Set Station ID message.

Authentication is the algorithm used to establish validity of the user. The CU checks validity of the RU by having the RU send a unique 48 bit MAC ID assigned to manufacture of the RU. The transmission of this MAC ID is done using the link management service of the physical layer. The CU then checks the received MAC ID against a table containing all the authorized MAC IDs.

Bandwidth management involves several issues. Bandwidth allocation is the process which enables control and allocation of bandwidth in the upstream channel. Static bandwidth management is performed to control bandwith reservation which is important to enable a RU to have guaranteed bandwidth so as to implement the quality of service requirement of the ATM protocol. Dynamic bandwidth management is performed in real time to manage bandwidth resources which include real time adaptation to shift bandwidth to other RUs that have increasing need from RUs that have excess bandwidth.

Bandwidth allocation is performed both statically and dynamically by the CU computer, and distribution of the decisions is made over the downstream channel. The allocation decisions are transmitted downstream with messages having a command type of the class Time Slot management. Time Slot management commands are additive in that only the difference over the last state of allocation is transmitted in the downstream channel when the allocation changes so as to minimize traffic. Because this causes the possibility that a RU will lose track of the state of timeslot allocation, an unsolicited state of bandwidth allocation is transmitted at a slow periodic rate in the preferred

embodiment. In alternative embodiments, either the entire allocation state can be transmitted each time the allocation changes, or only the differences may be broadcast with the CU supporting a function which can be invoked by any RU to transmit to that RU the entire allocation state when the RU loses the state information.

Static bandwidth allocation by a reservation mechanism is performed during connection establishment by an access request. The reservation informs the CU of the minimum sustained bandwidth required for the RU. Dynamic bandwidth allocation handles the burst nature of bandwidth requirements. A scheme is used by which the aggregate peak rate bandwidth allocated may exceed the maximum bandwidth possible. On the average, it is expected that only a small portion of the total available bandwidth shall actually be needed. In some embodiments, bandwidth allocated or reserved to one RU which does not actually need it at the moment may be temporarily assigned to another RU. The dynamic bandwidth manager requires the constant monitoring of the utilization of bandwidth by each RU. If usage drops below a certain threshold, bandwidth will be removed from the allocation of the RU. If traffic increases above another threshold, additional bandwidth will be allocated if there is available bandwidth according to the current priority allocation scheme in use.

Figure 36 illustrates a flow chart illustrating the dynamic bandwidth management process carried out by the CU computer and the computers in the RUs. This routine executes every 10 milliseconds and starts with a process of collecting bandwidth requests as symbolized by step 1100. This involves management traffic interchange between the RUs and the CU indicating the amount of bandwidth each RU wishes to have allocated to it. Next, as symbolized by block 1102, the CU reads the number of cells transmitted by each RU and transmitted to each RU. This information is counted in the ATM/SCDMA interface circuit 1009 in a manner to be described further below. This information reflects the number of cells that were transmitted to and from the RU in the last 10 milliseconds. The CU computer then uses these most current RU usage measurements to make a decision as to whether to suggest to the prioritization and allocation process symbolized by block 1108 to award more bandwidth or less to each RU. If the number of cells transmitted to and from a RU is less than a configurable percentage of a "minimum threshold" established relative to its current usage, the CU computer adds a deallocation suggestion message to the collection of bandwidth requests for that RU as symbolized by block 1104. If the current usage of a RU in the last 10 milliseconds is greater than a predetermined percentage of a configurable maximum threshold established relative to that RUs current allocation, the CU computer adds an additional allocation suggestion to the collection of bandwidth requests for that RU, as

symbolized by block 1106. This process is repeated for each RU. Next, in step 1108, the CU computer analyzes and establishes priorities for bandwidth reservations, bandwidth requests, and bandwidth allocation and deallocation suggestions generated in steps 1104 and 1106 for each RU in accordance with the current priority channel allocation scheme. The CU computer then decides how much bandwidth to award each RU and which channels to allocate to each RU to implement these allocation decisions. The CU computer then distributes those allocation decisions by downstream messages, as symbolized by step 1110.

Figure 38, comprised of Figures 38A, 38B and 38C, is a more detailed flowchart of the preferred embodiment of the processes of block 1108 and 1110 of Figure 36 showing how the allocation decisions are made and distributed. The process starts with block 1130 which represents the step of resetting a RU counter to zero to prepare for scanning all RUs for their requirements and actual usage. Next, in step 1132, the CU computer 1015 creates a table having a row for each RU, each row having multiple fields. Then, starting with the RU pointed to by the current contents of the counter, the CU computer reads certain data pertaining to that RU and writes each item of data to the appropriate field in the row which pertains to the RU currently pointed to by the RU counter. The fields in a row for each RU are illustrated in Figure 39. Step 1132 represents the process of reading and storing for the RU pointed to by the current contents of the RU counter the following items of data. Any pending bandwidth requests not already processed are read and stored in field #1. The current RU bandwidth usage in the form of the number of ATM cells transmitted to and received from that RU in the last 10 milliseconds is read, and that data is stored in field #2. The current RU allocation of channels, meaning the allocation awarded this RU during the last 10 millisecond period is stored in field #3. This data is transferred from the new RU allocation field #4 which stores data indicating how many channels this RU was awarded during the channel allocation calculation performed during the last 10 milliseconds;. Next, any new bandwidth or channel reservations made by this RU in access requests which have not already been processed are stored in field #5 in the "new" subfield. Field #5 also has an "old" subfield which stores the previous iteration reservation number which is needed to determine if channels have been borrowed temporarily from this RU during a previous iteration, as described below. The CU computer also checks field #6 to determine if this particular RU has privileges to make bandwidth reservation requests. Such privileges will be awarded to subscribers who pay higher rates. Finally, any allocation/deallocation suggestions made in steps 1104 and 1106 in Figure 36 are read and stored in field #7.

The RU counter is then incremented in step 1134, and the test 1136 is performed to determine if the RU counter has been incremented past the last existing RU. If not, step 1132 is performed again for the next RU pointed to by the RU counter. Finally, when all RU data has been gathered, path 1138 is taken to test 1140. Test 1140 determines whether the actual usage by any RU which has had channels borrowed during previous iterations from the channels the RU previously reserved to it (as indicated by a current allocation less than the old reservation number stored in the "old" subfield of field #5) has risen to a predetermined percentage of the reserved bandwidth. Actual bandwidth usage is calculated by multiplying the number of ATM cells transferred to and from the RU during the last 10 millisecond period times the number of bits in an ATM cell, the result divided by 10 milliseconds to give bits per second. Each channel in the old reserved channel number has a maximum bit transfer rate of 8 megabits per second. Therefore, the percentage of the reserved bandwidth actually used is calculated by dividing the actual number of bits per second by the quantity [number of channels reserved in the old reserved channels number times 8,000,000 bits/second/channel]. The percentage of the reserved bandwidth that will trigger a yes result in test 1140 is configurable in some embodiments and fixed in others. If this percentage is equalled or exceeded, path 1142 is taken to step 1144 where, in the preferred embodiment, enough channels are allocated to those RUs that have channels borrowed but which now need some of the reserved capacity back, to provide an allocation for those RUs which is the predetermined percentage above the actual usage. These allocated channels are then recorded in the new RU allocation fields #4 for each of the RUs that have had their allocations updated in this manner. Processing then proceeds to step 1146. Likewise, if test 1140 indicates that no borrowing of channels from reserved channels has occurred or that actual usage by any RU from which channels have been borrowed has not risen to the point where channels have to be given back, then path 1148 is taken to step 1146.

Step 1146 totals all reservation requests by all RUs that have a privilege to reserve, by adding all the numbers in the column defined by the "new" subfield in bandwidth reservation field #5 for all rows of the table with a "yes" indication in field #6. Next, test 1150 is performed to determine if the total number of reserved channels calculated in step 1146 is less than or equal to the 128 available payload data channels. If the answer is yes, step 1152 is performed to allocate to all RUs the number of channels each reserved, and those allocations are written into the new RU allocation fields #4 for each RU which has reserved channels. Those same reserved channels are written into the "old" subfield of field #5 for use in subsequent iterations to determine

if borrowing has occurred. The numbers in the "old" subfield of field #5 do not change until a new reserved number of channels is awarded. Likewise, the numbers in the "new" subfield of field #5 do not change until a new bandwidth reservation request is issued by a RU.

Because of the possibility that there are RUs that have made bandwidth requests that either are not reservations or the RU has made a reservation request but is not authorized to make such a request (in which case the reservation request is treated as an ordinary bandwidth request) and there is still available bandwidth, process proceeds from step 1152 to step 1153 representing the start of the request allocation routine shown on Figure 38C.

Returning to the consideration of test 1150, if it is determined that the total number of reserved channels calculated in step 1146 exceeds the number of available payload channels (normally 128 unless some portion of them have been allocated on a fixed or permanent basis to some RUs), then path 1154 is taken to test 1156 to determine if demand can be met by borrowing. Test 1156 examines the actual usage of bandwidth by each RU compared with reserved bandwidth by comparing the number in field #2 to the number in the "old" subfield of field #5 for each RU with a number in the "old" subfield of field #5. If the actual usage number for any RU with reserved bandwidth is low enough compared to the reserved bandwidth to justify temporarily borrowing some bandwidth, then path 1158 is taken to step 1160.

Step 1160 borrows only enough channels to fill the needed number of reserved channels determined in step 1146. The borrowing is done from RUs determined in test 1156 to have usage which is low enough compared to the amount of bandwidth reserved to justify the borrowing. The percentage of reserved bandwidth which is low enough to justify borrowing can be a fixed percentage in some embodiments or can be configurable in the preferred embodiment. The borrowed channels are allocated to RUs requesting reserved channels and which had actual usage in the last 10 millisecond period indicating the reserved channels are actually needed. After these allocations are made, the current RU allocation data in field #4 is updated for all RUs being awarded new channels and from which channels have been borrowed.

After step 1160 is performed to do any borrowing which is possible and necessary, step 1161 is performed to send downstream allocation messages to all RUs informing them of their new allocations in accordance with the data in the new allocation fields #4 in the table. Processing then returns to step 1130 on Figure 38A to start the process over again.

If test 1156 determines that no borrowing of reserved channels is justified, path

1162 is taken to step 1164. Step 1164 handles the overflow of reservation requests by allocating available channels to the highest priority RUs in the number each reserved until the supply of available channels is exhausted. The CU computer sends downstream messages informing the RUs that have had channels reserved to them as requested the channel numbers that have been reserved to them and updates the data in field #3 of the table and the "old" subfield of field #5 to reflect the new allocations and the corresponding reservations. Finally, step 1164 is completed by sending downstream messages to RUs that did not get the reserved channels that they asked for that system capacity is overbooked and to make their request again later. Processing then proceeds to step 1166 to wait for the next iteration and then returns to step 1130 on Figure 38A to start the process over again on the next iteration.

Referring to Figure 38C, there is shown a flow diagram of the processing for allocating bandwidth requests that are not in the nature of reservations. Step 1153 represents the start of this process. Step 1153 is reached from step 1152 on Figure 38B if reservation requests are less than total available bandwidth and reservations have been allocated as requested and there is leftover bandwidth or from step 1152 in the case where no reservation requests have been made and all available bandwidth is to be allocated according to regular bandwidth requests. The first step in this process is symbolized by block 1168 where the CU computer calculates how many payload channels are still available after any reservation allocations (or other allocations such as fixed allocations) are taken into account. Then, test 1170 determines if the number of available channels is zero. If the number of available channels is zero, step 1172 is performed to send allocation downstream messages to all RUs per the new RU allocation data in fields #4 in the table. In all downstream allocation messages, only the differences from the last allocation are sent to each RU.

If test 1170 determines that channels are available, step 1174 is performed. This step adds to the data in field #3 for each RU (the current RU allocation of channels) any suggested additional channel allocations generated in step 1104 of Figure 36 and subtracts from the data in field #3 for each RU any suggested channel deallocations. These totals are then stored temporarily in the new RU allocation fields #4 in the table.

Next, test 1176 is performed to total these proposed new allocations of nonreserved, available channels by adding up the numbers in fields #4 for RUs that have not reserved channels, as indicated by zeroes in the "old" subfield of field #5 and to test that total against the number of available channels. If the total derived by step 1176 is less than the number of available channels, the temporary allocation reflected in fields #4 is retained as the new allocation. In that case, path 1178 is taken to step 1180

where downstream messages are sent out to the RUs indicating their new allocations in accordance with the data stored in field #4 of both reserved and unreserved but awarded channels. In other words, these messages tell RUs that have reserved channels that their reservations have been accepted, and tell RUs that have not reserved channels but which have requested them that the number of channels they requested have been awarded to them.

If test 1176 determines that the total new allocation of unreserved channels in accordance with requests would exceed the total available number of channels, then path 1182 is taken to step 1184. Step 1184 revises the new allocation data in fields #4 to allocate the available, unreserved channels in a weighted manner per the received bandwidth requests and the allocation and deallocation suggestions in the order of priority of the RUs. In other words, the RU requests plus allocations and deallocations are honored in the order of priority of the RUs. However, in the preferred embodiment, this prioritization will save enough of the available channels to allocate at least one channel to each RU which made a bandwidth request. In alternative embodiments, the available channels may simply be allocated in accordance with the bandwidth requests from the RUs and suggested allocations and deallocations until the available channels are exhausted. In either type embodiment, downstream messages are formulated telling the RUs which channels have been allocated to them, or, if no channels have been allocated to one or more RUs, downstream messages may inform these RUs that system capacity has been temporarily exceeded and to try again later.

The priority channel allocation scheme is designed to ensure that RUs requiring bandwidth lower than their minimum guaranteed rate will get this bandwidth always. However, this means that while a RU is negotiating a connection setup, a connection may be refused if the aggregate minimum guaranteed bandwidth exceeds the bandwidth available. That is, the following mathematical relationship must always be maintained:

$$\sum_{VLI} B_{MAX} \leq B_{AVAIL}$$

There is however no restriction on the peak rate allocated, apart from the restriction that the maximum peak rate approved must be less than the available bandwidth. Figure 37 illustrates the operation of the bandwidth allocation scheme. The initial bandwidth allocation is made at station identification time indicated at 1112. Subsequently, the RU makes a request for additional bandwidth at 1114. Assuming the request is granted, an additional allocation represented by step 1116 is given. Thereafter, as actual bandwidth

used rises, additional allocations of bandwidth symbolized by steps 1118 and 1120 etc. are automatically given. This process of allocating and deallocating bandwidth continues as actual usage rises and falls until the bandwidth needs of that RU are satisfied.

DATA LINK LAYER

The data link layer provides the capability to transfer ATM cells between multiple end stations and the CU and vice versa. The hardware and software on the data link layer handles the following issues: ATM cell framing - such that the beginning and end of each ATM cell can be determined by both the RUs and the head end; addressing - providing the ability to send a cell to a specific RU, and to know from which RU a cell came; multicast/broadcast support - enabling the transmission of an ATM cell to multiple RUs simultaneously; interface to upper layers - provides a standard interface to upper layers.

Framing of ATM cells is as described above. Note that the downstream ATM cell format is the only cell format where the ATM cell needs to have two additional bytes of virtual link header in addition to the 53 bytes of a standard ATM cell. The virtual link header information is necessary to designate to which RU the ATM cell is directed. The upstream cell format needs no virtual link header information because all ATM cells are directed to the same place, i.e., the head end. However, both cell formats need "start of ATM cell coding" such as 9-bit bytes so that the 9th bits can be coded with a "start of ATM cell" code.

Referring to Figure 40, there is shown a block diagram for the preferred structure for the multiplexer/demultiplexer ATM/SCDMA interface or formatter 1009 in Figure 20. The circuits below dashed line 1200 handle data in the downstream path, while circuits above the dashed line 12 handle data in the upstream path although some circuits like the rate counters 1202 and the master CPU local bus 1204 are shared by both sets of circuits.

The downstream circuits receive Utopia + format ATM cells on bus 1206 from the SAR (segmentation and reassembly) controller 1002 in Figure 20. The data on bus 1206 takes the form of a TDM stream of ATM cells with one 55 byte ATM cell in each time slot. These ATM cells have 53 bytes of data with standard ATM cell header plus a two byte virtual link header identifying the particular RU to which they are directed. This virtual link information is added by SAR 1002 in Figure 20 from information received on bus 1003 in Figure 20 from a computer carrying out higher level central management and control functions at the head end for assigning virtual link IDs. The 55 byte ATM cells are received by a cell input controller 1208 and applied to the data inputs of a cell buffer memory 1210 which is one megabyte in depth in the preferred

embodiment. The cell input controller 1208 is a memory controller for memory 1210 which generates addresses to store all 55 bytes of each ATM packet. In the preferred embodiment, these addresses are sequential. The combination of cell buffer memory 1210 and cell input controller 1208 implement a pipeline memory stage to provide rate buffering between two circuits which may be processing data at different rates, e.g., the SAR and the SCDMA physical layer. If, for example, some peripheral coupled to the CU is pouring data into the SAR at a high rate, but the amount of bandwidth awarded to the virtual link on which the torrent of data is to be processed is temporarily too low to send out the incoming data as fast as it is arriving, the cell buffer memory 1210 temporarily stores the data until the physical layer catches up.

Each time a new ATM cell arrives on bus 1206, the cell input controller 1208 retrieves a new base address from a free cell buffers memory 1212. Memory 1212 stores the starting addresses of each ATM cell stored in cell buffer memory 1210 which has been read therefrom by a cell extractor circuit 1214 for transmission by the physical layer. The cell input controller 1208 then generates sequential offset addresses for each byte of the cell and writes the bytes into memory 1210. As the cell extractor reads each ATM cell's data out of memory 1210, it stores the starting address of the cell in free cell buffer memory 1212 via bus 1216. The cell input controller then retrieves a free base address at which to start storing the data from the next cell from the free cell buffer memory 1212 using bus 1218.

Link status table memory 1222 stores data regarding which cell buffers (a sequence of 55 byte storage locations in cell buffer memory 1210 will be called a cell buffer herein) in cell buffer memory 1210 actually contain an ATM cell which has not yet been read out of the memory for transmission by the physical layer. When the cell input controller 1208 writes an ATM cell into a cell buffer within cell buffer memory 1210, it writes the base address, i.e., the starting address of the cell buffer into link status table 1222 via bus 1224. The link number of the virtual link identified in the two byte virtual link header is also stored in link status table 1222.

The cell extractor/data framer 1214 determines which is the next ATM cell that needs to be read from the cell buffer memory 1210 by reading the output data from the link status table memory 1222. The cell buffer memory 1210 is managed as a FIFO, so the ATM cells can be processed by the physical layer in the order received from each source (with interleaving among sources) so that the order of the ATM cells from each source can be preserved. The link status table memory 1222 keeps a record of the order of reception of the ATM cells stored in the cell buffer memory 1210 for each virtual link and outputs the starting address of the next ATM cell that needs to be processed for

each virtual link. The cell extractor/data framer 1214 receives information from the master CPU 405 and its local bus 1230 regarding which timeslots on bus 1232 to which each virtual link has been assigned. The function of the cell extractor/data framer 1214, at its highest level, is, for every timeslot on bus 1232, to extract the next ATM cell in sequence which is directed to the RU assigned to the virtual link corresponding to the timeslot being filled. The ATM cell is extracted from cell buffer memory 1210, in the same order as received from the source which is sending data to the RU corresponding to the virtual link and timeslot being filled. The ATM cell so extracted is then disassembled and one byte from the ATM cell is placed on bus 1232 in its assigned timeslot. The next time a timeslot assigned to the virtual link to which the ATM cell so extracted is directed becomes available for filling, the extractor/data framer 1214 retrieves the next sequential byte from the same ATM cell previously extracted and places it in the timeslot. This process continues until the entire ATM cell has been transmitted to the RU to which that ATM cell is directed.

The cell extractor/data framer 1214 reads all the bytes of each ATM cell via bus 1220 in the proper order using addresses supplied on bus 1223 from the link status table and adds the 9th bit to each byte. The cell extractor/data framer 1214 encodes the 9th bits of the first 8 bytes of each ATM cell with a start code to show the downstream formatter where the ATM cell starts and to include CRC information for the ATM cell data so that errors can be detected and corrected.

Bus 1232 is coupled to time division multiplexing framer circuit 1234. The function and structure of this TDM framer circuit 1234 is to take the 9-bit bytes out of the cells and place them on bus 1011 coupled to the ASIC 1013 in Figure 20. The data on bus 1011 is organized as a time division multiplexed stream comprised of timeslots corresponding to the 144 channels of payload and management and control information. The TDM framer circuit places the 9-bit bytes of the ATM cells on bus 1011 aligned with the timeslot boundaries, and when there is no ATM cell data to send, generates idle cells and places them on the bus 1011. The TDM framer includes circuitry to take the data from cell extractor/data framer 1214 via bus 1232 in parallel format and place it on bit serial format bus 1011 with each byte aligned with a timeslot boundary and each timeslot on bus 1011 carrying 9 bits aligned with 9 bit times within the timeslot. ATM cell boundaries are not required to be aligned with frame boundaries. The idle cells have their 9th bits encoded with start codes at the beginning of each cell so as to keep the RU cell framing recovery circuitry in sync since ATM cell boundary synchronization is maintained using the downstream 9th bit data.

The cell input controller 1208 is coupled to the rate counters circuit 1202 by

bus 1240. Bus 1240 is used by the cell input controller to update information in the rate counter memory 1202 each time an ATM cell is stored in the cell buffer memory 1210. Bus 1240 is used to increment the count of ATM cells sent to whatever RU is assigned to the virtual link number in the virtual link header of the ATM cell just stored. A similar connection, bus 1241 is used by a cell output controller 1243 for a similar purpose in updating ATM cell numbers received in the upstream data from each RU.

The rate counters circuit is a memory which stores the number of ATM cells received from each RU and the number of ATM cells transmitted to each RU during each 10 millisecond period between bandwidth reallocation calculations. The rate counter circuit is read by the master CPU 405 via bus 1204 during the bandwidth reallocation calculations to determine actual bandwidth usage by each RU. The rate counter data is indexed by virtual link number, which the computer 405 translates to particular RU IDs since computer 405 knows which RUs have been assigned to which virtual links at all times.

Upstream data from the RUs arrives as a time division multiplexed, bit-serial data stream organized into 128 timeslots on bus 1011, each timeslot carrying 9 bits. Bus 1011 has two separate data paths, one in each direction. The time division multiplexed data is received by TDM framer 1242. The incoming data on bus 1011 is a stream of bits which must be reassembled into the 9 bit bytes of the ATM cells from which the data originated. The TDM framer 1242 recovers the bit and byte boundaries and reassembles the bits into 9-bit bytes and outputs them on bus 1244. Frame boundaries are recovered with the help of Frame and Superframe control signals from the ASIC 1013 on lines 1246 and 1248, respectively. The Frame signal is activated at the start of each frame boundary. The TDM framer 1242 then counts out 9 bit times for each timeslot thereby recovering the bit and byte boundaries and counts out 128 timeslots. Each 9-bit byte recovered from the bitstream on bus 1011 is tagged by the TDM framer with a timeslot ID indicating for each 9-bit byte the timeslot from which it came. The timeslot ID or number for each byte corresponds to the virtual link or channel, i.e., the SCDMA code used to transmit that byte. The channel or code used identifies the RU from which the byte came since the timeslot/channel allocation assigned to each RU every 10 milliseconds is known. TDM framer 1242 corresponds to deframer 470 in Figure 3 and F^{-1} circuit 856 in Figure 12.

The stream of 9-bit bytes on bus 1244 is received by data framer 1246 and reassembled into 53 byte ATM cells. The data framer 1246 uses the timeslot ID tags on each byte to determine from which RU each byte came. This is done by using the timeslot ID for each byte as an index into a timeslot assignment table 1248 via bus 1250. The

timeslot assignment table stores data correlating each timeslot number to the virtual link number in accordance with the bandwidth/channel allocation decisions made by the CU central management and control process carried out in computer 1015 or some other computer. The channel allocation decisions are input to the timeslot assignment table via master CPU local bus 1204 each time a reallocation is made so as to update the table data. When the table is presented with a timeslot number, it returns a virtual link ID number indicating the RU from which the byte originated. The data framer uses the virtual link numbers to get bytes into the appropriate ATM cells and uses the start codes encoded into the 9th bits coupled with the fact that it is known that there are only 53 bytes to each ATM cell after the virtual link header is stripped to find the ATM cell boundaries. Only conventional 53 byte ATM cells are needed at this point because for the upstream data, there is only one destination and, at this point in the stream, the source RU is known by virtue of the timeslot number tag on each byte.

The data framer 1246 keeps track of the state of completion of each cell using a link status table in memory 1260. The link status table stores the current address in cell buffer memory 1254 in which was stored the last byte retrieved from bus 1244 for the ATM cell under construction for each virtual link number. In other words, the data framer is receiving interleaved bytes from many different RUs, all belonging to different ATM cells. The data framer must keep track of which ATM cell is under construction for each virtual link number, and where in that ATM cell under construction the next received byte which is part of that ATM cell is to be stored. To derive the address where the next byte for any particular ATM cell is to be stored, it is only necessary to retrieve and increment the "current address" stored in the link status table in memory 1260 for the virtual link over which this ATM cell has been transmitted. The data framer 1246 uses the link status table in memory 1260 to keep track of the process of constructing multiple ATM cells being transmitted from multiple sources so that bytes retrieved from timeslots are stored in the correct ATM cell and in the correct sequence within each ATM cell. The data framer 1246 reads and updates the data in memory 1260 via bus 1262.

After storing each new byte in an ATM cell buffer in memory 1254 at the address pointed to by the link status table, the data framer 1246 determines if the byte just stored completes the ATM cell being constructed by checking to see if the address in which it stored the byte is the 53rd sequential address in the ATM cell from the base address at which storage of data for this ATM cell started. If so, then the data framer retrieves the next free ATM cell buffer base address from the free cell buffers memory 1252 and starts construction of a new ATM cell there when the next byte from the same

RU arrives. Also, upon completion of an ATM cell, the data framer 1246 writes the base address of the completed cell into a full cell buffers memory 1264 via bus 1266. The address data stored in the full cell buffers memory 1264 is used by the cell output controller 1243 in reading ATM cells out of cell buffer memory 1254 for distribution to the SAR. Each full ATM cell stored in a cell buffer in memory 1254 is tagged with the virtual link number over which the data was transmitted. In the preferred embodiment, each RU is assigned a constant virtual link number and only the codes and timeslots used to carry data for that virtual link change from one bandwidth allocation to another.

The data framer 1246 like the TDM framers 1234 and 1242 and the cell extractor/data framer 1214 as well as the cell input controller 1208 and 1243 can be implemented either as hardware or software or some combination of the two.

After an ATM cell has been reassembled in memory 1254 from the data in the timeslots on bus 1011, it is formatted into a Utopia + format for output to the SAR. A two byte virtual link header is added to each of the 53 byte ATM cells stored in cell buffer memory 1254 so as to convey information to the SAR 1002 in Figure 20 as to where the data came from. That process is carried out by the cell output controller 1243. This controller reads the base address of a completed ATM cell from the full cell buffers memory 1264 via bus 1268. The ATM cell and its virtual link tag stored beginning at the base address so retrieved is then read out and a two byte virtual link header is added to the cell. The modified cell is then placed in a timeslot on bus 1270 and transmitted to the SAR. The cell output controller then writes the base address in memory 1254 of the ATM cell just read into the free cell buffers memory 1252 for use by the data framer later in storing a new cell in an unused buffer. After extracting an ATM cell and placing it on bus 1270, the cell output controller 1243 also updates the rate counter information in rate counter memory 1202 via bus 1241 to indicate another ATM cell has been received over a particular virtual link. The data in the rate counters memory 1202 is used for both billing and for the adaptive bandwidth allocation algorithm.

Referring to Figure 41, there is shown a flow chart of operations by the data framer 1246. The process starts at block 1272 by setting the timeslot counter to 0 to synchronize to the frame boundary. In other words, the process of block 1272 is not performed until the Frame signal on line 1246 in Figure 40 is activated indicating that a new frame of 128 timeslots is arriving on bus 1244. Then the entire process of Figure 41 is performed for each timeslot on bus 1244.

The first byte in the first timeslot on bus 1244 is retrieved in the process symbolized by block 1274. The virtual link number for this timeslot is retrieved in the

process symbolized by block 1276 by using the timeslot count as an index into the timeslot assignment table stored in memory 1248 in Figure 40. Block 1278 symbolizes the process of verifying, during the first 8 timeslots only for this particular ATM cell, that the 9th bit for the particular timeslot whose 9-bit byte was just read from bus 1244 is the proper bit for the 8 bit start code stored in the 9th bits of the first 8 bytes. The data framer knows which byte of the ATM cell each RU is currently transmitting because the CU knows when each cell starts from the start codes and then keeps a running count for each virtual link as to which byte in the ATM cell is the current byte. This count is maintained by the data framer by updating the data in the link status table stored in memory 1260. This process of block 1278 is done to verify synchronization at the start of every ATM cell. If the 9th bit is not what it is supposed to be, some error or loss of synchronization has occurred. In that case, error recovery processing which is not part of the invention may be performed, or the byte may simply be ignored. One type of error recovery that can be used is to send a message in the downstream data telling the RU that has lost synchronization to resynchronize.

Block 1280 symbolizes the process that is carried out if the 9th bit is found in block 1278 to be what it is supposed to be for the particular timeslot being processed. Block 1280 represents the process of accessing the link status table using the virtual link number as the index and retrieving the current address. The current address in cell buffer memory 1254 is then incremented, as symbolized by block 1282. The byte read from bus 1244 is stored in the cell buffer memory 1254 at the address pointed to by the incremented current address as symbolized by block 1290.

Test 1292 determines if the current address is greater than or equal to 53. This means that the offset address (the "address" used in step 1290 to store the byte just stored) from the base address in which is stored the virtual link number of this ATM cell has reached 54 or more. It takes 54 memory locations to store an ATM cell of 53 bytes, because the virtual link number is stored in the base address location. If that is the case, the last byte stored was the last byte in the ATM cell under construction in the cell buffer memory 1254, and the ATM cell has been completely reconstructed from the data in the timeslots on bus 1244 in Figure 40. In such a case, path 1294 is taken to the process symbolized by block 1296. If the ATM cell under construction is not yet complete, path 1298 is taken to test 1299 where it is determined whether the timeslot count is equal to 128. If the timeslot count has reached 128, the last timeslot in the frame has been processed, and processing is vectored back to block 1272 where the timeslot counter is reset to zero. If all 128 timeslots of the frame have not been processed, the timeslot counter is incremented in block 1301, and processing proceeds

to block 1274 to get the next data byte from the next timeslot on bus 1244.

Returning to the consideration of test 1292, if the last byte in the ATM cell under construction has been stored for the virtual link identified in block 1276, block 1296 transfers the base address of the completed ATM cell to the full cell buffers FIFO memory 1264. This insures that the ATM cell just completed will be read out from the cell buffer memory 1254 by the cell output controller 1243 in the order it was received.

Next, to prepare for storing of the next ATM cell from the same virtual link, the process symbolized by block 1298 is performed to get the base address of an empty ATM cell buffer in cell buffer memory 1254. This is done by accessing the free cell buffers memory 1252 and pulling an address off the stack. Block 1300 symbolizes the process of storing the base address retrieved in this manner as the new current address. Then, the virtual link number retrieved in block 1276 is stored in this base address, as symbolized by block 1302. Processing then proceeds to test 1299 where it is determined whether the current timeslot being processed is the last timeslot in the frame.

Referring to Figure 42, there is shown a flow chart of the processing carried out by the cell output controller 1243 to retrieve completed ATM cells from the cell buffer memory 1254 and transfer them to the SAR. First, test 1310 is performed to determine if the full cell buffer FIFO 1264 has any addresses in it. If the FIFO is empty, the path 1312 is taken to stay in a holding pattern until an address appears in the full cell FIFO buffer. If there is an address in the full cell FIFO buffer, the process symbolized by block 1314 is performed to transfer the ATM cell to the SAR by reading the base address in memory 1254 of the ATM cell from the full cell FIFO buffer 1264 and using it to access the bytes of the ATM cell and the virtual link number starting at the base address. Block 1314 also represents the process of retrieving the virtual link number stored at the base address for the ATM cell being transferred and appending it as the two byte virtual link header in front of the 53 bytes of the conventional ATM cell the bytes of which are stored in sequential addresses in memory 1254 starting at the base address. The 55 byte ATM cell constructed in this manner in step 1314 is then placed in a timeslot in a Utopia + format data stream on bus 1270 in Figure 40 and transferred to the SAR.

Block 1316 represents the process of writing the base address of the ATM cell just transferred to the SAR into the free cell buffer memory 1252 as an indication that this cell buffer is now available for overwriting with data from the next cell. Finally, the process of block 1318 is performed to increment the rate counter data for this virtual link to show that another ATM cell has been received on this virtual link.

Processing then returns to test 1310 to start processing for the next cell.

Referring to Figure 43, there is shown a block diagram of the formatter 1030 in Figure 20. The purpose of the formatter in each RU is, for the downstream data: to receive a stream of downstream data including ATM cells directed to that RU, detect that fact, strip off the 2 byte virtual header and reconstruct the ATM cells and output the ATM cells in Utopia format. The purpose of the formatter in each RU is, for the upstream data: to receive a Utopia format TDMA stream of ATM cells directed to the CU from the peripherals coupled to the RU, add a 9th bit to each byte to encode ATM cell start codes, fill in any gaps with idle cells so cell timing synchronization will not be lost, fragment the ATM cells into their constituent bytes and output them as a TDMA stream.

Referring first to the downstream circuitry, the ASIC 1024 demodulates, descrambles and despreads the incoming RF signals and generates a time division multiplexed stream of 9-bit bytes in 128 timeslots on bus 1320. A byte recovery circuit receives this bit stream and is responsible for reassembling the 9-bit byte transmitted during each of the 128 timeslots. The byte recovery circuit 1322 receives a Frame signal on line 1324 from the ASIC which is activated by the ASIC at the beginning of each new frame. The byte recovery circuit then starts counting out timeslots by counting 9 bits per timeslot and assembling the first 9 bits as the first timeslot byte and the next 9 bits as the next timeslot byte. The byte recovery circuit receives a bit clock signal on line 1326 from a bit clock recovery circuit (not shown) and uses this bit clock signal to establish windows to recover the individual bits. The bytes on bus 1320 will not be scrambled and sequential bytes will all be from the same ATM cell until an ATM cell boundary occurs.

The collection of 9-bit bytes recovered by byte recovery circuit 1322 is output on bus 1328 to a cell timing and recovery circuit 1330. The cell timing and recovery circuit uses the 9th bits in the stream of 9 bit bytes to find and synchronize to the ATM cell boundaries. Once the ATM cell boundaries are recovered, the 55 bytes of each ATM cell are assembled and are buffered (buffering occurs in some embodiments and not in others depending upon the relative processing speeds of the stages of the formatter).

The ATM cells so assembled are then passed to a comparison circuit 1332. The comparison circuit compares the two byte virtual link header of each ATM cell to the virtual link number of the RU in which the formatter resides. If there is a match, the two byte virtual link header is stripped off the ATM cell, and a 53 byte ATM cell is transmitted on bus 1334 to output circuit 1336. The output circuit presents the 53 byte ATM cell in a timeslot of Utopia format TDMA bus 1338 coupled to SAR 1034 in Figure 20.

The upstream data path starts with a Utopia format ATM cell, time division multiplexed data stream on bus 1339 from SAR 1034 in Figure 20. This data stream is received by a Utopia interface circuit which functions to add a 9th bit to each byte in each ATM cell. The Utopia interface circuit 1340 also serves as a memory controller for a small cell buffer FIFO 1342. When a complete ATM cell has been encoded with the 9th bits, the cell is stored in the cell buffer FIFO 1342. In some embodiments, this is done as in the case of the multiplexer/demultiplexer 1009 shown in Figure 40. In these embodiments, the Utopia interface circuit 1340 accesses a free cell buffer memory 1344 and retrieves a base address of a free cell buffer in FIFO 1342. The 53 byte ATM cell of 9 bit bytes is then stored in 53 sequential addresses starting with the base address of the cell. The base address of the cell is then written by the Utopia interface circuit into a full cell buffer FIFO 1346.

To maintain cell boundary synchronization in the upstream data path, idle cells are transmitted during gaps when no real data is waiting to be sent. These idle cells have 9-bit bytes, and the first 8 bytes of each cell are encoded with a start code identical to the start codes in real ATM data cells. These idle cells are generated by idle cell generation circuit 1348.

A selector switch memory controller circuit 1350 functions to retrieve the ATM cells out of cell buffer FIFO 1342 when there are ATM cells stored there, and retrieves idle cells from idle cell generation circuit 1348 when there are no real ATM cells waiting to be sent. The selector switch memory controller retrieves the base address of any ATM cells stored in cell buffer FIFO 1342 from the full cell buffer FIFO memory 1346 and uses the base address of each cell to read the bytes of the cell. When a full ATM cell has been read from the cell buffer FIFO memory 1342, the base address of the buffer storing the cell is written by the selector switch memory controller into the free cell buffer 1344. The selector switch memory controller switches between sources of cells only on 53 byte boundaries.

The ATM cells read by the selector switch memory controller 1350 are available for transmission on bus 1352 to a byte extractor circuit 1354. The function of this byte extractor circuit is to pull a byte out of the selector switch memory controller each time the ASIC 1024 activates a Clear To Send signal on line 1356. The ASIC receives ATM cell data from the peripherals in the form of a 128 timeslot TDMA 9-bit byte stream on bus 1358. Although there are 128 timeslots on this bus, only the timeslots awarded to this RU are "active". In other words, this RU can transmit a byte only during the one or more of the 128 timeslots allocated to it by the bandwidth allocation algorithm. The byte extractor implements the media access control algorithm by

extracting a byte from an ATM cell via bus 1352 only when the Clear To Send signal on line 1356 has been activated by the ASIC. This signal will be activated only when a timeslot allocated to this RU is occurring on bus 1358. The byte so extracted is transmitted on bus 1360 to a TDMA formatting circuit 1362. This circuit places the byte so extracted on the bus 1358 in the timeslot allocated to the RU and in bit synchronization with the bit times of the timeslot.

Lower Overhead Optimized Embodiment With Reduced ATM Header Size

One of the advantages of using standard ATM cells with 5 byte headers in an embodiment like that shown in Figure 20 is that the full logical connection ability of the ATM protocol can be used. The 5 byte ATM header has 24 bits of addressing capability called the VPI/VCI bits. These 24 bits allow for up to approximately 16 million different logical connections at each RU. The advantage of this is that multiple peripherals at each RU may each have multiple processes running each of which has its own logical address. However, 16 million is more logical connections than each RU needs. The 5 byte ATM header represents about 10% of the total number of bytes in an ATM cell. Many of the bits are present for purposes not really needed in the CATV SCDMA environment in which the invention operates. Accordingly, the 5 byte standard ATM cell headers represent wasted bandwidth. The presence of 16 million possible different logical connections also unnecessarily complicates the design of the formatters and multiplexer/demultiplexer circuits in the RU and CU modems, respectively. Therefore, optimization of the system by better bandwidth utilization and less circuit complexity can be achieved by reducing or eliminating the size of the ATM cell headers.

However, ATM cells with 5 byte headers could also be used in the embodiment of Figure 44 if the software and hardware of the formatters inside the RU modems and the multiplexer/demultiplexer circuit inside the CU modem is configured properly to handle and generate these headers as well as generate the additional 2-byte virtual link header designated 1070 in Figure 28. This provides a system where each RU is coupled by an Ethernet LAN to as many peripherals as necessary and provides the full ATM functionality. Although, in the preferred embodiment, Ethernet LANs are used to illustrate the teachings of the invention, the teachings of the invention are equally applicable to other types of networks also, so any reference to Ethernet in this specification or the appended claims should be understood as a reference to any local area network protocol.

In the preferred embodiment, the class of cable modems represented by Figure 45 are designed to use optimized ATM cells that have only 2 byte headers in the downstream direction and no headers at all in the upstream direction. In these

embodiments, the small 2 byte headers in the downstream ATM cells identify the logical channel on which each ATM cell is to be placed (to which RU the ATM cell is directed). The Ethernet address identifying the particular peripheral coupled to the RU to which the data is to be directed after arriving at the RU is contained within the payload data.

The two byte headers used in the downstream ATM cells are the 16 least significant bits of the VPI/VCI bits in a standard 5 byte ATM cell header. These 16 bits are used to identify the RU modem to which the 48 byte payload is to be sent. Figure 46 illustrates the optimized downstream ATM cell with 2 byte header 1391 identifying the virtual link, i.e., the single logical channel serving one RU, on which the 48 bytes of payload data 1393 is to be transmitted.

Figure 47 illustrates the optimized upstream ATM cell as including no header and only a 48 byte payload 1395. The "bytes" in the payloads of each of the upstream and downstream ATM cells are 9-bit bytes. The 9th bits are encoded with special codes which give the system the ability to distinguish between idle cells which are transmitted to maintain synchronization when there is no real data to transmit, first optimized ATM cell in a packet, a normal optimized ATM cell somewhere in the middle of the packet and the last cell in a packet.

Referring to Figure 44, there is shown a block diagram of the optimized header preferred embodiment of a system species within the genus of the invention. The embodiment of Figure 44 is what will be referred to as an optimized embodiment because substantial bandwidth has been saved by reducing the size of the ATM cell header in the downstream direction and eliminating it altogether in the upstream direction. In the embodiment of Figure 44, the CU router equipment and interfaces to various service and wide area networks are represented by block 1370. In a preferred internetworking embodiment where the RU can be coupled to wide area networks having TCP/IP transport protocols through an interface at the head end, block 1370 represents a standard router with a SAR built in to parse the inbound IP packets into standard 5-byte header downstream ATM cells and to package inbound 5-byte header ATM cells from the CU modem 1372 and the RU modems into IP packets for delivery to the router. The circuitry inside block 1370, communicates with the CU modem 1372 using the industry standard bidirectional OC3 ATM protocol in a TDMA stream of standard 5 byte header ATM cells on bus 1374. The OC3 ATM protocol is the same as a Utopia format ATM cell TDM stream in that complete ATM cells are transmitted during each timeslot except that line drivers are present which can drive the ATM cells over much greater distances and standard TDMA techniques are used to account for the larger distances involved such as making the timeslots are wider than the ATM cell bursts to allow for propagation delays

and using reference bursts to establish carrier and bit timing plus transmission of a unique word to establish an accurate time reference for each frame, said reference bursts being transmitted at the beginning and end of each frame. The Utopia protocol is a chip to chip data stream protocol which has a very short range and need not use these TDMA techniques.

The basic idea behind optimization is to use only the part of the standard 5-byte ATM cell header needed to define the destination RU and eliminate the rest thereby eliminating needless bandwidth consumption for transmission of portions of the standard 5-byte header which are not needed. An ATM cell header has 24 bits of VPI/VCI data which defines the final address to which the cell is to be delivered. Optimization involves limiting each RU to only one logical channel and using the least significant 16 bits of VPI/VCI fields of the standard header to define that single logical channel and eliminating the rest of the header. The multiple peripherals coupled to each RU or the multiple data sources coupled to the CU modem that wish to send data to a particular RU have their data time division multiplexed into the single logical channel connecting that RU to the CU modem. Packets from different sources are reconstructed using the source addresses in the packets as will be explained in more detail below.

For optimization in the upstream direction, each ATM cell has no header at all since all upstream ATM cells are going to the same place, i.e., the CU modem and since the source RU can be identified by the timeslot in which the data from the RU is output by the TDMA framer 1242 in Figure 40.

Details of the Optimized Downstream Process

A summarization of the process of sending optimized ATM cells downstream over an HFC cable plant is as follows. In the embodiment of an optimized system shown in Figure 44, the CU modem 1372 does all the processing necessary to take the ATM cells arriving in the downstream direction on bus 1374 in standard OC3 ATM format (5 byte standard ATM cells) and reduce the header size to the 2 bytes which identify the logical channel and RU to which the data is directed and transmit the data. Transmission of the downstream data can be by any standard transmission technology such as QPSK, TDM, QAM-64, QAM-256, or encoding the payload data using sequential or randomly assigned SCDMA spreading codes, etc. It is the upstream data which is sent by SCDMA in the preferred embodiment to implement the virtual links needed to transmit ATM cells from multiple sources to a single CU modem over a shared HFC cable plant. In alternative embodiments, TDMA, FDMA or other multiplexing techniques can be used to establish the virtual links on the shared media.

The 5 byte standard ATM cell header in each ATM cell in the OC3 ATM format data

stream on bus 1374 includes 3 bytes (24 bits) of address data called the VPI/VCI bits. It also includes one 8-bit byte checksum and 8 additional bits 4 of which are reserved for future expansion of the protocol and the other 4 of which are used for miscellaneous ATM functions which are not relevant to the invention. In the optimized embodiment of Figure 44, the CU modem, for downstream ATM cells strips all of the bits of the standard 5 byte ATM cell headers except for the 16 least significant bits of the 24 VPI/VCI bits. These 16 VPI/VCI bits identify the particular logical channel into which each ATM cell is to be transmitted in the downstream direction. The 48 9-bit bytes of payload data from the ATM cell is then appended to this new 2 byte header and the optimized downstream cell, as shown in Figure 46, is then output in a TDM stream inside CU modem 1372 to the downstream transmitter. If the downstream data is sent by spread spectrum techniques, the SCDMA spreading process can work as previously described herein or any other conventional spread spectrum technique can be used.

In alternative embodiments, the TDM buses in the CU and RU can be omitted. In these alternative embodiments, the optimized 50 9-bit byte downstream ATM cells can be transferred from the formatter 1385 in Figure 48 to the transmitter 1013 via some other method than by a TDM stream on bus 1387. The same is true for the upstream direction. For example, the formatter function of associating particular 9-bit bytes to be transmitted on particular virtual links with timeslots assigned to that virtual link could be incorporated into the transmitter ASIC. In such an embodiment, if both the upstream and downstream transmitters were using SCDMA, the function of the formatter would be modified to "associate" the bytes to be transmitted on each virtual link with the CDMA spreading codes to be used in transmitting those bytes, i.e., make sure bytes destined for transmission over a particular virtual link have their spectrums spread using the proper CDMA spreading codes assigned to that virtual link. In the case of downstream transmission by other than SCDMA, the function of the formatter in the alternative embodiment is to receive the standard 53 8-bit byte ATM cells, strip off the header bytes and replace them with a two byte optimized header, add a 9th bit to each byte and encode the last cell, idle cell, normal cell and start code information into these 9th bits, and parse the optimized downstream ATM cells so generated into parcels of bits for transmission by the transmitter using whatever modulation scheme it is using and deliver these bit parcels to the transmitter using any known data transfer mechanism. In still other alternative embodiments, the formatter may not add the 9th bits but instead may encode the start code cell framing data and cell type information in one or more additional bytes of header. As long as these functions are performed by the formatter, it is unimportant what circuitry is used to perform these functions, where

the formatter circuitry is located or the manner in which the data to be transmitted is transferred into or out of the formatting process. Since downstream data is broadcast to all RUs, logical coupling of data to particular virtual links by the formatter need not be done.

The same statements made about the formatter in the CU generally hold true for the formatters in the RUs such as formatter 1030 in Figure 45 except that in the upstream direction, SCDMA or at least CDMA is used in the preferred embodiment to enable a unique virtual link to be established between the CU and each RU over the same media used by all the RUs. The upstream data is not broadcast, and each RU has only one virtual link. The purpose and function of the formatter in each RU is therefore to do the header optimization, add the 9th bits and encode them with cell framing and cell type auxiliary data (or to add the auxiliary data encoded into the 9th bits into one or more header bytes), and to associate all the incoming 9-bit bytes from the host RU with the particular CDMA or SCDMA spreading codes assigned to that RUs virtual link.

The preferred embodiment uses a QAM-64 transmitter in the CU modem for the downstream data transmission. QAM-64 transmitters are well known in the literature and embodiments thereof are described in the reference texts incorporated herein by reference. The details of the QAM-64 transmitter are not critical to the optimization invention. Basically, the QAM-64 transmitter receives the optimized downstream ATM cells and divides each cell into sequential smaller packets of 7 bits of data and 1 CRC bit. Each of these 8-bit bytes are divided into two 4-bit nibbles and used to modulate the amplitude of one of two carriers of the same frequency but 90 degrees out of phase. The receivers in the RU modems 1382 and 1380 reverse the process to detect the 8-bit bytes, error detect and reassemble the ATM cells. Each RU then looks at each 2-byte ATM cell header and discards any ATM cells that are not directed to that RU. If TDM is used, the ATM cells are parsed into sequential smaller groups of bits, calculating CRC on the smaller groups and transmitting the smaller groups and CRC in sequential timeslots.

The resulting RF signals are output on CATV HFC link 1000 in whatever frequency band is reserved for downstream data. Upstream data from the RUs 1380 and 1382 is transmitted in SCDMA encoded RF signals in another frequency band as is the case for the other non-optimized embodiments disclosed herein.

A more detailed discussion of the downstream process follows. Figure 48 is a more detailed block diagram of the circuitry inside blocks 1370 and 1372 in Figure 44. Figures 49A, 49B and 49C together comprise a flowchart of the optimized downstream process. The details of the circuitry and software that do the processes detailed in both the downstream processing flowcharts as well as the upstream processing flowcharts are

not critical to the invention, and any circuitry and software structure that can do the packet transformations described herein will suffice to practice the invention.

In the embodiment shown in Figure 48, the CU equipment block 1370 includes a conventionally designed router 1371 which couples to a TCP/IP protocol wide area network such as the Internet, represented by line 1373. The router need not be located in the CU, and could be physically located elsewhere and coupled to the CU by a network. TCP/IP packets (or any other wide area network packet format if the wide area network is not the Internet) arrive on bus 1373 to be processed by the router. The IP packets take the form shown in Figure 50A, and are converted by a software process in the router 1371 to the packet formats shown in Figures 50B and 50C. The IP packet symbolized by Figure 50A includes a header which has at least an IP destination address 1401, an IP source address 1403, and a payload data section 1404. IP packets are less than 1.5 kilobytes long. Other fields such as CRC may also be present but are not shown as they are not relevant. The router software uses the IP destination and source addresses to look up Ethernet domain destination and source addresses and appends an Ethernet destination address 1406 and an Ethernet source address 1408 to the front of the IP packet, as symbolized by Figure 50B. This process is symbolized by block 1500 in Figure 49A. In the more general case, the incoming packet can be from any wide area or local area network coupled to the CU (referred to in some of the appended claims as the first network), and the destination address data in the incoming packet header corresponds to the address of the destination node in the address space of the first network, the destination node being coupled to an RU by a second network such as an Ethernet LAN etc.

The lookup tables in the router bind Ethernet domain addresses and virtual link identifiers to IP addresses. These lookup tables are built by the router in a unique feature of the system disclosed herein by use of the DHCP protocol. The IP software layers in the peripherals coupled to the RUs are programmed to control the peripheral so as to invoke the DHCP (Dynamic Host Configuration Protocol - an industry standard protocol) each time they are powered up to request an IP address. The router software monitors this DHCP message traffic requesting an IP address which the IP software layer in the peripheral controlled the peripheral to send to a DHCP server and the reply message to the peripheral from the DHCP server assigning an IP address to the peripheral. The router software then uses the IP address and the Ethernet domain (or other network domain) source address of the peripheral on which the IP software layer which made the request is running to build its routing tables in memory. These tables are used to look up the Ethernet domain addresses used to construct the Ethernet header

symbolized by blocks 1406 and 1408 in Figure 50B.

The software in the router, after appending the Ethernet domain addresses, then appends an additional header field 1410 defined by national standard RFC 1483 (as that standard existed at the time this application was filed) to the packet of Figure 50B to arrive at the packet format symbolized by Figure 50C. These RFC 1483 bytes (which are also added in the upstream direction) are referred to in the claims as "unique marker bits". This process is symbolized by block 1502 in Figure 49A. In the claims, the RFC 1483 bytes are referred to as predetermined unique marker bits since other unique bits could also be used in the case where the second network coupled to the RU was other than an Ethernet. Header field 1410 with RFC 1483 bytes simply labels the packet as an Ethernet packet and signals where it starts. In the claims, the packet shown in Figure 50B is referred to as the first modified packet. It is the packet format shown in Figure 50C which is referred to in the claims as the second modified packet, and it is this packet which is sent on bus 1377 in Figure 48 to the SAR 1375 segmentation and reassembly process. The router communicates bidirectionally with SAR 1375 via bus 1377.

The SAR takes packets of the format shown in Figure 50C and adds sufficient pad bytes 1412 in Figure 50D to make the total number of 8-bit bytes after addition of the pad bits and CRC bits to be an integer multiple of 48, the number of bytes in an ATM cell payload. The SAR 1375 then calculates CRC-32 error detection bits on the whole packet including the pad bits, and appends them as CRC field 1414 to complete the AAL5 packet structure shown in Figure 50D. This process is symbolized by block 1504 in Figure 49A. The packet structure shown in Figure 50D is referred to in the claims as the third modified packet, and it will be understood by those skilled in the art that the CRC bits are calculated on the packet including the pad bits and that is the meaning of the phrase "calculating error detection bits" in the claims.

Next, the SAR parses each AAL5 sequence packet like that shown in Figure 50D into sequential, conventional 5-byte header, 48-byte payload ATM cells. This process involves taking each packet of the format of Figure 50C and chopping it into sequential 48-byte ATM cell payload sections, like payload section 1393 in Figure 46, starting from the RFC 1483 field 1410 and proceeding sequentially all the way to the end of the packet. To each 48-byte payload section, a standard 5-byte ATM cell header is added by the SAR. This process is symbolized by block 1506 in Figure 49A.

This 5-byte ATM cell header added in step 1506 includes a 4-bit GFC field (not relevant), an 8-bit VPI field, a 16-bit VCI field (these VPI and VCI fields contain the destination address of the RU to which the ATM cell is directed), a 4-bit CLP/PTI field

(only the last bit of the 3-bit PTI field which is relevant) and a HEC field which is a checksum on the first 4 bytes of the ATM cell header. The standard ATM cell 5-byte header also encodes some other useful information. If VPI/VCI are both 0, then the cell is an idle cell carrying no payload. If the last bit of the PTI field is a one, then this ATM cell is the last cell in the AAL5 packet structure of Figure 50D. This idle cell and last cell information will be later encoded by formatter 1385 into the two 9th bits of the two 9-bit header bytes of the optimized downstream ATM cell. In the claims, the last cell, idle cell, normal cell and start code data may be referred to as auxiliary data.

Each "byte" of either an upstream or downstream ATM cell is actually 9-bits. The 9th bits are added by the formatter in the preferred embodiment, but can be added by the SAR in other embodiments. In the claims these 9-bit bytes may be referred to as "component bit groups", but this term is also intended to cover groups of bits that are less than 9 bits long such as in embodiments wherein the TDM streams to the transmitters do not contain full 9-bit bytes in each timeslot assigned to a virtual link but some lesser or greater number which are then used by the transmitter for encoding and transmission.

The two 9th bits of the two byte header in the downstream optimized cell can be encoded to define four possibilities. Three of these codes are used to indicate whether each ATM cell is a normal cell, an idle cell or the last cell in the AAL5 packet. The SAR uses the header information in the AAL5 packet to generate the VPI/VCI information in the 5-byte normal ATM cell header to be appended to each 48 9-bit byte payload and generates the rest of the standard 5-byte header from its tables of data, the rest of the header not specifically used in the optimized process generally being zeroes or don't care values. The content of the rest of the 5-byte header outside the VPI/VCI field and the last bit of the PTI field is not relevant and will be stripped off by the formatter in the CU modem anyway, so these values could be hardwired to any values. The SAR calculates the HEC checksum field.

The SAR encodes the last bit in each PTI field of each ATM cell which is not the last cell in the AAL5 sequence with a zero indicating it is not the last ATM cell in the AAL5 sequence, and encodes the last bit in the PTI field of the ATM cell which is the last cell in the AAL5 sequence with a one indicating it is the last cell. This process is represented by block 1508 on Figure 49B. The standard 53-byte ATM cells encoded with last cell information by the SAR are symbolized by Figure 50E.

In some embodiments, the SAR is also coupled by bus 1379 to another interface circuit 1381 which is coupled to data sources which are local to the CU and may be coupled to the SAR by a local area network. Downstream data arriving on bus 1379 also

includes destination address data identifying the destination peripheral or destination software process to which the data is directed. Typically, the other interface 1381 may be an Ethernet controller coupled to an Ethernet segment 1383 coupling the interface 1381 to a hub or switch. Processing of these Ethernet packets or other data streams into standard 53 byte ATM cells proceeds generally as described above except that no lookup of the Ethernet domain addresses bound to IP addresses is necessary.

The SAR outputs the standard 5-byte header ATM cells generated from the AAL5 sequence packets (generated from the inbound IP or LAN packets) as a TDMA OC3 data stream on bus 1374 to multiplexer/demultiplexer (hereafter formatter) 1385 of the CU modem 1372 in Figure 48. This process is symbolized by block 1510 in Figure 49B.

Each standard 53-byte ATM cell composed by the SAR is transmitted in one timeslot on bus 1374, so formatter 1385 knows the ATM cell boundaries. The formatter 1385 needs to encode the data to be transmitted so that the RU receivers can locate the ATM cell boundaries and locate the last ATM cell in an AAL5 sequence packet so that the Ethernet packets can be reassembled and so that the formatter can determine when an ATM cell is an idle cell. The formatter also serves to optimize the downstream ATM cell by stripping out all the 5-byte header information from each ATM cell received on bus 1374 (after error checking the header using the HEC field) so as to save only the least significant 16 bits of the VPI/VCI fields as an optimized two-byte downstream header. The last bit of the PTI field encoding the last cell information is also preserved by encoding it into the 9th bits of the 2-byte header of the downstream optimized ATM cell. In other words, the formatter 1385 generates the optimized 2-byte downstream ATM cell header by encoding the normal cell, last cell, idle cell information into two 9th bits which are appended to the two 8-bit bytes comprising the least significant 16 bits of the VPI/VCI field. This process is symbolized by block 1512 in Figure 49B.

In alternative embodiments, the ATM cell boundary and cell type information (referred to sometimes herein as the auxiliary data) may be encoded into downstream ATM cells by means other than encoding it into the 9th bits. The manner in which this information is encoded and transmitted downstream is not important so long as the receiver can locate and use the auxiliary data to find the ATM cell boundaries and reassemble the Ethernet packets encapsulating the IP packets at the RUs. In the claims, these 9th bits or the additional header bits added to each ATM cell to encode the auxiliary data are referred to as "additional bits". For example, one alternative embodiment could encode the cell boundary and cell type data into a third byte of an optimized 3-byte downstream ATM cell comprised of 51 8-bit bytes. Therefore, in the claims, the phrase

"adding a plurality of additional bits to each ATM cell ... and encoding said additional bits with auxiliary data" is intended to read on both encoding the auxiliary data in the 9th bits or in one or more additional header bytes as well as other possible ways of encoding the auxiliary data into the ATM cell.

The optimized downstream two-byte header enables the RU receivers to determine whether the ATM cell is directed to that RU, whether to discard the payload because it is an idle cell and to locate the end of the AAL5 sequence for purposes of assembling Ethernet packets. The formatter 1385 also encodes the first 8 9th bits of the first 8 bytes of the payload section of the downstream ATM cell with a start code so to enable the receivers at the RUs to determine where the ATM cell starts. This is symbolized by block 1514 in Figure 49B. In formatter 1385, it is the VPI/VCI bits of the standard 5-byte header which are used to look up a virtual link number assigned to the destination RU in a mapping table kept in the formatter at the head end. This virtual link data maps Ethernet domain and IP destination addresses to virtual link numbers. The virtual link address data is then placed in the 2-byte header of each ATM cell generated at the CU to make sure the ATM cell gets to the correct RU modem.

The optimized downstream ATM cells are then parsed by the formatter into 9-bit bytes and output in sequential timeslots on TDM bus 1387 to downstream transmitter 1013. The transmitter 1013 can be any conventional QAM, QPSK, TDMA, CDMA or any of the SCDMA embodiments disclosed herein or any known conventional transmitter technology. The process of transmitting these 9-bit bytes over the HFC cable plant using any conventional digital data transmission technology is represented by block 1518 in Figure 49B.

The optimized system uses a two level addressing scheme and a mapping between each logical channel and the assigned RU for that channel. The two byte header in the downstream optimized ATM cell identifies the single logical channel upon which the data is to be transmitted, and this single logical channel corresponds to a single one of the multiple RUs. The Ethernet address of the particular process or peripheral at the RU to which the payload data is to be directed once it arrives at the RU is included as several bytes in the payload data.

After modulation of the downstream data onto the RF carrier or carriers, the resulting RF signal is output on line 1017 to an RF up/down converter 1018 for conversion to the proper downstream frequency band and output on HFC link 1000.

The RF signals transmitted downstream by the CU are received at each RU modem, of which modems 1380 and 1382 in Figure 44 are typical. Each of these modems has the structure illustrated in the block diagram of Figure 46 and operates generally in the

manner described above with reference to Figure 20 except that each modem is coupled to an Ethernet hub through an Ethernet controller (1402 in Figure 45) instead of directly to one or more peripherals. Each of the peripherals such as digital VCR 1386 is coupled to the Ethernet hub. Specifically, modem 1380 in Figure 44 is coupled by Ethernet link 1381 (either 10BaseT, 10Base2, etc.) to an Ethernet hub 1384. The hub is coupled by separate Ethernet links to three peripherals illustrated by digital VCR 1386, personal computer or workstation 1388 and video server 1390. All the circuitry inside dashed line 1392 is located at the premises of customer #1. Likewise, all the circuitry inside dashed line 1394 is located at customer premises #2. A similar Ethernet setup with different peripherals exists at customer premises #2 and is coupled by Ethernet link 1396 to modem 1382. The hub is coupled to a PC or Macintosh 1398, a Sun Workstation 1400 and a Videophone 1402.

The only difference between the way modems 1380 and 1382 operate versus the operation described for the customer premises modem illustrated in Figure 20 is that in Figure 44, the SARs inside the modems at the customer premises translate between Ethernet packets and ATM cells whereas the RU SAR in Figure 20 may construct ATM cells from serialized or parallel data streams received directly from a peripheral.

Referring to Figure 45, a description of the downstream processing carried out by the RU modems 1380 and 1382 in Figure 44 will be given. One structure difference between the RU modem of Figure 45 and the RU modem shown in Figure 20 is that each modem of the type shown in Figure 45 has an Ethernet controller 1402 which couples the modem to the Ethernet hub via an Ethernet link such as link 1381 in Figure 44.

The details of the processes in the RU modem which operate on the downstream data are as follows. Receiver 1405 in Figure 45 receives RF signals from the HFC network 1000, does carrier recovery and other timing recovery and demodulates and decodes these signals using conventional processes. The 9-bit bytes recovered in this manner are output in sequential timeslots on the downstream portion of TDM bus 1026 for input to the formatter 1030. This process is symbolized by block 1520. The TDM transmission of downstream 9-bit bytes on bus 1026 replicates the TDM stream of bytes on the downstream portion of bus 1387 to the transmitter 1407 in the CU modem of Figure 48. The optimized ATM cells of 50 9-bit bytes which comprise the original AAL5 packet format of Figure 50D are received by the formatter as the byte stream illustrated in Figure 50F. The bytestream on bus 1026 in the downstream direction comprises a plurality of optimized 50-byte downstream ATM cells arranged end-to-end in sequential timeslots on bus 1026.

The RU modem formatter 1030 recovers the ATM cell boundaries by looking for

the start code of each ATM cell encoded into the 9th bits of the first 8 payload bytes and reassembles the 50-byte optimized downstream ATM cells, as symbolized by block 1522 in Figure 49C. The RU modem formatter then examines the two 9-bit byte headers of the optimized downstream ATM cells and discards any cells whose headers indicate they are not directed to this RUs logical channel, as symbolized by block 1524 in Figure 49C. The resulting optimized downstream ATM cells, filtered on the 2-byte header to eliminate ATM cells not directed to this modem are represented by Figure 50G.

These optimized downstream ATM cells, filtered by logical channel, are output on bus 1408 to the SAR as a Utopia stream, also as symbolized by block 1524. In an alternative embodiment, the formatter can reconstruct a full 5-byte ATM header from the data in the two byte optimized header and data in tables in the formatter and send the reconstructed, standard 53-byte ATM cells to the SAR as a Utopia TDM stream on bus 1408.

The RU SAR 1406 in Figure 45 uses the ATM cells received by the SAR on bus 1408 to recover the packet boundaries of the AAL5 sequence packet of Figure 50D. This is done by locating the RFC 1483 bits located somewhere in the 48 byte payload portions of the received ATM cells. These RFC 1483 bits indicate the start of the Ethernet AAL5 sequence packet of Figure 50D. The end of the AAL5 sequence packet is found by the SAR by examining all the 9th bits of the 2 byte headers of the optimized downstream ATM cells or by examining the PTI field last bit in the case where conventional 5-byte ATM cell headers were reconstructed by the formatter before transmission to the SAR. These 2 9th bits of the 2 byte headers will hereafter be referred to as the last cell codes even though these 2-bit codes also carry other information such as whether the ATM cell is an idle cell or normal cell.

When the SAR finds a header with a last cell code coded to indicate this ATM cell is the last ATM cell comprising the complete AAL5 packet of Figure 50D, the process of reassembly of the AAL5 sequence packet is carried out, all of the above processing being symbolized by block 1526 in Figure 49C. The details of the AAL5 packet reassembly are as follows. First, the SAR checks for errors in the AAL5 packet by performing an error detection calculation on the contents of the AAL5 packet used on the CU side to compute the CRC bits included with the last ATM cell payload in the AAL5 packet. The results of this error detection calculation are then compared with the CRC bits included with the last cell, and if they match, no errors have occurred. If there is a mismatch, some error has occurred, and the ATM cells comprising the AAL5 packet are discarded. Since the Ethernet and/or IP packets are sequentially numbered, the Ethernet or IP processes in execution at the destination machine will detect the loss of this packet and carry out an

appropriate recovery protocol such as asking for retransmission etc. If there were no errors in the AAL5 packet, the SAR strips off all the 2-byte or 5-byte headers from the ATM cells. Then, the SAR strips off the pad bits and the CRC bits from the last ATM cell and concatenates all the ATM cell payload sections of the ATM cells starting with the ATM cell payload section which contains RFC 1483 bits, including all the ATM cell payload sections in the middle and ending with the remainder of the ATM cell payload section which included the last cell code to regenerate an AAL5 sequence packet which looks like the packet shown in Figure 50H. The resulting packet is stored in memory 1404 in Figure 45 via a DMA transfer over bus 1405 and a pointer to the packet pointing to the start of the Ethernet header section 1406/1408 is transmitted to the Ethernet controller.

The Ethernet controller 1402 in the RU removes the packets like that shown in Figure 50H from memory starting with the Ethernet header pointed to by the pointer received from the SAR, and sends the packet shown in Figure 50I out over the Ethernet LAN such as LAN 1396 in Figure 44 for normal Ethernet processing. Eventually the packet reaches the destination node designated in the Ethernet header 1406/1408. This process is symbolized by block 1528 in Figure 49C. This destination node, like all the other destination nodes has software processes in execution thereon symbolized by the software process architecture diagram shown in Figure 50. Each node has an Ethernet process layer 1450, an IP layer 1452 above that, a TCP layer 1454 above the IP layer and an application layer 1456 at the top of the software process stack. The Ethernet layer process 1450 is where the Ethernet headers are examined to determine if the Ethernet packet is addressed to this node. If so, the Ethernet header is stripped off, and the remainder of the packet is transferred to the IP layer by sending a pointer to where the packet is stored in memory or by some other interprocess transfer mechanism. The IP layer examines the IP header and transfers the packet through the TCP layer 1454 via an appropriate interprocess transfer mechanism to the appropriate process on the application layer 1456 which is bound to the IP address given in the IP header 1401/1403. The IP layer also handles the situation where the error check previously done indicates the IP packet has an error in it. The IP layer carries out conventional error handling protocols in the case an error has occurred to cause the IP packet with the error in it to be resent or other suitable recovery processes.

Details of the Upstream Processing to Send Optimized ATM Cells Via SCDMA Over HFC

A general summary of the upstream process carried out by the RU modem is as follows. The Ethernet controller receives upstream Ethernet packets and temporarily

stores them by DMA transfer into memory 1404 and takes Ethernet packets reassembled by a SAR 1406 and transmits them out to the Ethernet hub. SAR 1406 takes Ethernet packets out of memory 1404 by DMA transfer at whatever rate the current allocation of upstream bandwidth to the RU supports. The SAR also stores reassembled downstream AAL5 type packets in memory 1404. Thus, memory 1404 serves as a buffer against different data rates on the Ethernet controller and SAR sides. The SAR disassembles the Ethernet packets and uses the header and payload data information to reassemble standard 5-byte header upstream ATM cells which are output in Utopia format on bus 1408 to formatter 1030. The formatter operates on the standard ATM cells to strip off the 5-byte headers of upstream ATM cells and places the payload data of each cell in the appropriate timeslot or timeslots on TDM bus 1026 assigned to the logical channel of the virtual link corresponding to the RU of which the formatter is a part. The Ethernet address of the source peripheral from which the data originated is encoded in payload 9-bit bytes and placed in the payload. The virtual link/RU identity from which the ATM cell originated is identified by the timeslots in which the bytes from the ATM cell arrived at the CU. The upstream TDM organized data is output on bus 1026 to SCDMA transmitter 1024 which operates as previously described to spread the energy of the data using SCDMA spreading codes assigned to the logical channel for this RU. The formatter does the mapping between the number of SCDMA codes and the timeslot numbers currently assigned to the RU and its logical channel. Note that the one logical channel per RU may be implemented using more than one SCDMA code in a dynamic fashion such that the number of SCDMA codes used per logical channel is dynamically allocated depending upon the bandwidth needs of the RU. The transmitter 1024 receives update messages giving the changes in current timeslot and SCDMA code assignment from the RU CPU via bus 1401 and informs the formatter of these assignments by messages on bus 1403. In alternative embodiments, the formatting process can be performed inside the transmitter so no bus 1401 is needed. These messages are used to update a mapping table in the formatter. This table is consulted when placing payload data into the timeslots on the TDMA bus 1026.

The ASIC outputs an RF signal on line 1022 to the RF up/down converter circuit 1020 in Figure 20 for translation to the proper frequency band for the upstream data direction.

Now, more detail on the upstream process will be presented. IP packets prepared by the IP layer software in the peripheral devices coupled to the RU modems propagate down through the Ethernet layer and have Ethernet headers appended thereto and are transmitted over the LAN to the RU modem.

The Ethernet packets from the LAN are received by Ethernet controller 1402 and are stored in memory 1404 via DMA transfers. The Ethernet controller passes pointers to the locations of the start of Ethernet packets in memory 1404 to the SAR 1406.

SAR 1406 retrieves the Ethernet packets from memory, adds RFC 1483 header bytes to the front of the Ethernet packets. The SAR also generates AAL5 padding bits and adds them to the end of the Ethernet packet just before the spot where CRC-32 bits to be calculated are to be placed. The SAR then calculates CRC-32 error detection bits on the entire collection of header, payload and pad bits. These CRC bits are added to the end of the Ethernet packet. The resulting packet in an IP format packet encapsulated within an Ethernet packet and looks like the packet shown in Figure 50D. The number of pad bits added is set such that the number of bits in the packet is an integer number of 48 bytes. This SAR processing is symbolized by block 1530 in Figure 52A.

Next, the SAR parses the packet into multiple 48 byte ATM cells starting with the first of the RFC 1483 bytes and ending with the last CRC bits. Then, the SAR generates a standard 5-byte ATM cell header for each 48 byte payload by using data identifying the virtual link assigned to the RU to generate the VPI/VCI field and using configurable data to fill in the other fields in the standard ATM cell header except for the PTI field and the HEC field. The last bit of the PTI field is encoded with the last cell information to identify the 48 byte payload that includes the last 48 bytes of the packet retrieved from memory. Normal cell and idle cell information is also encoded into the PTI field. The HEC field is then calculated. The resulting 53 8-bit byte standard ATM cells are then transmitted to formatter 1030 as a Utopia TDM stream, one standard 53 8-bit byte ATM cell per timeslot. This processing is symbolized by block 1532 in Figure 52A.

The formatter then uses the VPI/VCI virtual link identifier data in the header of each ATM cell to look up the SCDMA codes currently assigned to that virtual link from a code assignment table that is kept up to data by code assignments flowing from the CU CPU through downstream message traffic to the RU CU and then to the SCDMA transmitter via bus 1401 and to the formatter 1030 via bus 1403. This timeslot/code assignment information controls which timeslots in the TDM stream on bus 1026 in which bytes from each cell are placed since the timeslots on TDM bus 1026 map to corresponding SCDMA codes. In alternative embodiments, where no TDM bus is used between the formatter and the SCDMA transmitter, such as where the formatter is included within the SCDMA transmitter, the VPI/VCI header information is used simply to control which SCDMA codes are used to spread the spectrum of the data stream for this virtual link, and TDM bus 1026 and code assignment bus 1403 are eliminated. In such an embodiment, the Utopia data stream on bus 1408 is simply extended to the SCDMA transmitter 1024,

and the receiver 1405 is coupled to the formatter located in the transmitter, the formatter then outputting downstream data via the Utopia data stream on downstream direction of bus 1408 to SAR 1406.

In the preferred embodiment, the formatter 1030 also adds a 9th bit to at least some bytes of each ATM cell, preferably all of them. These 9th bits are encoded with auxiliary data by the formatter. Specifically, the formatter encodes the 9th bits with cell framing start codes to indicate where each ATM cell boundary lies in the data stream. The 9th bits are also encoded by the formatter with the last cell, idle cell and normal cell information from the PTI field to enable the last ATM cell in an Ethernet packet to be located so that the Ethernet packet can be reassembled at the CU. In alternative embodiments, instead of using 9th bits for 48 9-bit byte upstream optimized ATM cells, the start codes, last cell, idle cell and normal cell information can be encoded in a one or two byte header preceding each 48 byte payload.

The formatter then strips off the 5-byte header from each standard upstream ATM cell received in the Utopia format datastream on bus 1408 to generate optimized upstream 48 9-bit byte ATM cells. The formatter, in the preferred embodiment, then parses each cell into individual 9-bit bytes and transmits the bytes sequentially to the SCDMA transmitter 1024 in the timeslots assigned to this RU's virtual link on the upstream path of TDM bus 1026. For example, if in a first frame, timeslots 3,9 and 12 are assigned to this RU's virtual link, the first three 9-bit bytes of the first optimized upstream ATM cell will be transmitted to the SCDMA transmitter 1024 in timeslots 3, 9 and 12. If, in the next frame, timeslots 5 and 36 are assigned to this RU's virtual link, the next two 9-bit bytes in sequence in the first optimized upstream ATM cell will be transmitted to the SCDMA transmitter in timeslots 5 and 36. In alternative embodiments, other methods of getting the 9-bit bytes to the SCDMA transmitter can also be used other than a TDM bus. For example, the 9-bit bytes (or 8-bit bytes where the 9th bits are replaced by one or more header bytes) could be sent to the SCDMA transmitter seriatim via a parallel or serial bus, and the SCDMA transmitter (CDMA transmitters could also be used where lower traffic volume is present and the higher crosstalk from lack of frame synchronization can be tolerated) could then take the received bytes and encode them with the proper SCDMA spreading codes assigned to the RU's virtual link and transmit them.

The SCDMA transmitter 1024 receives the information regarding which timeslots of each frame are dedicated to this RU's logical channel from its local microprocessor 1028 (not shown) which receives the information from the CU in management and control messages issued in response to bandwidth requests transmitted

to the CU by the RU. This processing of parsing the ATM cells into the individual components (referred to in the claims as "first bit groups"), which, in the preferred embodiment are 9-bit bytes, and transmitting these bytes (referred to in the claims as "second bit groups") to the SCDMA transmitter is symbolized by block 1536 in Figure 52A. Note that block 1536 assumes that the auxiliary data is encoded in 9th bits and that these 9th bits are transmitted to the SCDMA transmitter via a TDM bus. In the alternative embodiments discussed above, block 1536 represents the process of adding the additional bits in whatever form they are added, encoding them with auxiliary data, parsing the resulting modified ATM cells into second bit groups which may be complete 9-bit or 8-bit bytes or some smaller or larger group (referred to in the claims as "second bit groups") and sending them to the SCDMA transmitter by any known data transfer mechanism.

The SCDMA transmitter 1024 then spreads the spectrum of the upstream 9-bit bytes using the SCDMA or CDMA spreading codes mapped to the timeslots in which the bytes arrive from the formatter. The SCDMA transmitter 1024, in the preferred embodiment, interleaves and scrambles the data from the 9-bit bytes prior to spreading the spectrum thereof. Any CDMA or SCDMA or other multiplexing transmitter will work to practice the upstream transmission invention, but the synchronous CDMA transmitter disclosed herein with code diversity, equalization training, ranging to establish and maintain frame synchronization between the diverse RUs and scrambling and interleaving of the raw data prior to spreading is preferred because of its higher data carrying capacity and better signal-to-noise performance. The resulting spread spectrum data is then QAM modulated and transmitted over the HFC to the CU modem although other modulation schemes could also be used.

The SCDMA receiver 1409 then does clock and carrier recovery and recovers the individual 9-bit bytes from each RU's logical channel and places them into the timeslots on TDM bus 1387 in Figure 48 assigned to that RU's logical channel. The formatter 1385 receives these bytes from each RU in the timeslots on bus 1387 assigned to that RU. The formatter knows from which RU each 9-bit byte originated by virtue of the timeslot in which the byte is transmitted on TDM bus 1387 from the receiver to the formatter. Note that the essential function performed by the receiver to aid the formatter and SAR to reconstruct the packets that were transmitted by each RU is to essentially "tag" each byte with virtual link identifier information identifying the particular RU from which the byte originated. In the preferred embodiment using a TDM bus, the tagging with virtual link identifier information is done by placing bytes from specific RUs in timeslots assigned to those RUs. In alternative embodiments where no

TDM bus 1387 is used between the CU transceiver 1013 and the formatter, this tagging will have to take another form. Any method of identifying which bytes came from which RUs will be acceptable to practice the invention. In the claims, this process of getting the recovered bytes from the receiver to the formatter with identifying information is expressed by the phrase, "receiving the modulated carrier(s) and recovering the 9-bit bytes that were sent by said RU and sending each 9-bit byte to a formatting process along with information indicating from which RU each said 9-bit byte was transmitted". The process carried out by the SCDMA receiver 1409 is symbolized by block 1538 in Figure 52B. Of course in some embodiments, the TDM bus is eliminated, so the data recovered by the receiver is transmitted to the CU formatter by other conventional data transfer mechanisms such as parallel or serial format buses using any know protocol.

The formatter 1385 demultiplexes the time division multiplexed data on TDM bus 1387 in the manner described above with reference to Figure 40. Basically, the formatter reassembles 48 byte ATM cells and stores the ATM cells from each RU into an area of cell buffer 1254 in Figure 40 dedicated to that RU. The formatter uses the 9th bits to find the cell boundaries and then records the values of the first two 9th bits and strips all the 9th bits off prior to storing the cell in the cell buffer. Once a complete 48 8-bit byte ATM cell is present in the cell buffer, a pointer to the cell is sent to the cell output controller 1243. The cell output controller adds the standard 5-byte header of standard ATM cells using information derived from the timeslots in the TDM stream in which the bytes were transmitted as reflected in the area in cell butter in which the cell was stored. More precisely, the cell output controller knows which modem the cell came from by virtue of which portion of the cell buffer the cell was retrieved from, this cell buffer area having been selected on the basis of which timeslots in which the bytes arrived. The cell output controller then constructs the standard ATM cell 5-byte header for each cell by setting the GFC field to 0, by setting the VPI field to the preconfigured Y value (established upon setup of the system in a configuration process), by setting the VCI field to X (X is a value which is explained below in conjunction with the discussion of block 1904 of Figure 53A), by setting the value of the PTI field using the data encoded in the first two 9th bits recorded earlier, and calculating the HEC field. The output controller sends the standard 53 byte ATM cells so recovered as an OC3 TDMA stream on bus 1374 in Figure 48 to the SAR 1375. This processing is symbolized by block 1540 in Figure 52B.

The SAR 1375 receives each standard 53 byte ATM cell, error checks the headers using the HEC field and strips off the 5 byte cell headers retaining the VPI/VCI and PTI fields for use in reconstructing the IP packet. The SAR then finds the packet boundaries

of the packet shown in Figure 50D using the RFC 1483 bytes and the last cell encoding in the PTI fields of the ATM cell headers and concatenates the 48 byte payloads of the ATM cells into a reconstruction of the packet sequence shown in Figure 50D. The CRC bits are then used to check the packet for errors and it is discarded if there are errors. The RFC 1483 bytes in the header are stripped off by the SAR as are the pad bits and the CRC bits to leave a packet having the format of Figure 50I. This packet is transmitted to the router process 1371 on bus 1377 for IP processing and transmission to its destination somewhere on the wide area network. This process is symbolized by block 1542.

VLI To IP To MAC Address Binding Process

An important part of the ability to send and receive ATM cells from a wide area network such as the internet is the ability to convert the IP addresses of internet packets to VLI address data in ATM cell headers. The problem is address mapping. That is, with peripherals such as personal computers coupled to the various RUs powering up and their IP software layer entities controlling them to request new IP addresses and with these same peripherals being switched off and relinquishing their IP addresses, the IP address to VLI virtual link identifier to MAC layer address mapping is constantly changing. To enable peripherals coupled to many different RUs to talk to devices and process coupled to the CU by a wide area network such as the internet, a method of assigning IP addresses to this constantly changing array of peripherals coupled to the wide area network through the HFC network and keeping track of the mapping between these IP addresses and the VLI and MAC layer addresses that must be used to access them must be devised. A process has been devised to resolve this problem, and that process is described below with reference to the flow charts of Figures 53A and 53B.

Referring to Figures 53A and 53B, there is shown the process which occurs at the time of an RU power up event to assign a VLI identification number to that RU and bind that VLI to an IP address. Block 1900 represents the process of the RU powering up and establishing communication with the CU as previously described herein. The RU then sends the CU a request message in the command and control channels requesting assignment of a virtual link identification number for use. The CU modem computer assigns an unused VLI (Virtual Link Identification) and send a downstream message on the command and control channels telling the RU what its VLI is, as symbolized by block 1902. Assume that the VLI assigned to the new RU is X.

Next, the CU sends a message to the router telling it there is a new modem and asking the router to record in its routing tables a communication channel to that modem having an ATM virtual link identity having a VPI value of some predefined Y and having a VCI value of X, the same number assigned to the RU as its VLI. This process is symbolized

by block 1904. The VCI value of X is essentially the destination address.

Next, a computer process in execution on a computer coupled to the RU generates and sends a DHCP broadcast message requesting assignment of an IP address, as symbolized by block 1906. This broadcast message includes the Ethernet address of the computer which sent it and is sent over the virtual link assigned to the RU to which the computer is coupled. The message is broadcast by the CU and is heard by the router and is forwarded by the router to a DHCP server coupled to the router, as symbolized by block 1906. The DHCP server receives the message, assigns an IP address to the computer which requested it, and sends back a DHCP message assigning an IP address to the computer having the Ethernet address in the original message, as symbolized by block. This message includes the Ethernet address of the computer which requested the IP address. The router receives this reply message and updates its routing tables to bind the newly assigned IP address to the virtual link assigned to the modem to which the computer which generated the original request is coupled and to the Ethernet address of that computer, as symbolized by block 1910.

A further optimization of the system includes the ability to "extend" the presence of the router from the CU to the RUs by configuring the MAC address of the Ethernet controller in each of the cable data modems of the RUs to that of the router. Prior works have either configured the controller to promiscuous mode, and thus required bridging or routing functionality. The optimization of configuring the MAC address of the Ethernet controller in the cable data modem to that of the router enables the presence of multiple Ethernet devices at the RU without the need for expensive bridging/routing functionality in each RUs cable data modem.

AN ALTERNATIVE EMBODIMENT USING ANY MULTIPLEXING TO ESTABLISH UPSTREAM VIRTUAL LINKS

Referring to Figure 54, there is shown a block diagram for a CU modem for an alternative embodiment wherein any other form of multiplexing is used in the upstream direction to establish the virtual links needed to support ATM cell transmission over shared media 1000. Figure 55 is a block diagram of an RU modem for the alternative embodiment in which the CU modem of Figure 54 is used. Circuit elements having the same reference numbers as used to identify circuit elements in Figures 45 and 48 serve generally the same purpose in the circuits of Figures 55 and 54.

Referring to Figure 55, the RU modem receives upstream local area network data packets via LAN controller 1402 from local area network 1381 and stores them in memory 1404 and notifies SAR 1406 of the base address of each packet. Alternatively, the data may be received in any other manner and stored in memory 1402. SAR 1406

retrieves the packets from memory, computes CRC bits and adds them to the packet and adds a sufficient number of pad bits so that the resulting number of bits is an integer multiple of 48 8-bit bytes. The SAR then parses the packet so modified into a plurality of 48 byte ATM cells and adds a standard 5 byte ATM cell header to each ATM cell. The SAR 1406 uses virtual link information to construct the VPI/VCI field, and uses last cell, normal cell and idle cell information for each cell to construct the PTI field. The SAR also calculates the HEC field and adds it. The resulting ATM cells are transmitted to the formatter 2000 via bus 1408, however the transmission of the cells does not necessarily have to be by a TDM Utopia stream.

The formatter adds start code information to each ATM cell such that the beginning and end of each ATM cell can be located by the CU modem. This can be by adding 9th bits to some bytes, adding header bytes etc. If the 9th bit technique described earlier is used, the data in the PTI field in the header is used to encode the 9th bits with last cell/normal cell/idle cell information as well as the start codes if the optimized smaller header is to be used. If optimized headers are not to be used, the 9th bits are encoded only with start codes. If optimized headers are to be used, the formatter then strips off the unnecessary portions of the 5 byte ATM cell header from each cell (all 5 bytes are unnecessary in the upstream direction) after using the VPI/VCI field contents containing the logical channel ID (also referred to herein as a virtual link) for this RU to associate each ATM cell with the proper logical channel.

In the embodiment shown in Figure 55, the logical channels, that is the virtual point-to-point link between each RU and the CU are established using any form of multiplexing transmitter 2002. The function of the formatter in associating each ATM cell with the proper virtual link is to get data from each ATM cell generated by a particular RU to the multiplexing transmitter for upstream transmission to the CU over the virtual link identified in that cell's VPI/VCI field. Any way that the formatter 2000 informs the multiplexing transmitter 2002 on which virtual link to transmit the bytes from each ATM cell will suffice for purposes of practicing the invention. If the multiplexing transmitter 2002 is a TDMA transmitter, the formatter 2000 parses each ATM cell (either optimized or not optimized) into individual bytes (9-bit bytes if the 9th bit technique is in use) and places the bytes into timeslots of a TDM bus 1026 assigned to the virtual link belonging to the RU. The multiplexing transmitter then takes the ATM cell bytes from the timeslots and transmits them on the proper virtual link identified by the timeslot number. In a TDMA multiplexing transmitter 2002, this would involve using the data from each byte to modulate one or more RF carriers during the timeslot on the shared media 1000 assigned to the virtual link of the RU. If the

multiplexing transmitter 2002 is an FDMA transmitter, the transmitter would take each byte out of its timeslot on bus 1026 and use the data thereof to modulate a carrier of the frequency assigned to the virtual link associated with that timeslot. If a multitone transmitter were to be used for transmitter 2002, the transmitter would perform an inverse Fourier transform on a plurality of frequency components to generate a composite output signal which would be used to modulate an RF carrier. Each of the frequency components would represent one virtual link, so the data from one RU would be used to control the magnitude of one frequency component at a frequency assigned to that RU as its virtual link. The signals from each of the inverse Fourier transformations in the multitone transmitters would be summed for transmission over the shared media.

Any type of multiplexing transmitter can be used for transmitter 2002 so long as it allows multiple RUs to simultaneously send data to the CU without interference with each other such that the shared media looks to each RU like a point-to-point connection. The formatter in each RU can associate the individual bytes of each ATM cell with that RU's virtual link for the multiplexing transmitter in other ways also. For example, the formatter can tag each byte with its virtual link ID and store it in memory shared with the transmitter. When the transmitter is ready to send data, it can retrieve a byte from memory and send it on the appropriate virtual link. Any way that the formatter 2000 and multiplexing transmitter 2002 cooperate to get data from each ATM cell transmitted on the CU via the proper virtual link falls within the teachings of the invention, and the above examples are not intended to limit the scope of the structure to those specific examples, but simply to suggest to those skilled in the art two straightforward ways of accomplishing this. Likewise, the above cited examples for the multiplexing transmitter 2002 are not intended to be an all inclusive list, and any transmitter that can establish a virtual point-to-point link over a shared media will suffice. For example, any of the multiplexing transmitters known in, typically, the satellite communication, telephony and cellular telephony arts can be used to establish the virtual links with known modifications to accommodate for the problems peculiar to whatever the shared media is.

Referring to Figure 54, after the upstream transmissions reach the CU receiver, an demultiplexing receiver 2004 capable of extracting the multiplexed data on each virtual link is used to separate out the data transmitted from each RU. This demultiplexing receiver 2004 can be, typically, any known type of receiver from the satellite, telephony or cellular arts which is capable of separating out data transmitted by the multiplexing transmitters 2002 on the various virtual links. Clock and carrier recovery can be by any conventional means.

The data parsed out by demultiplexing receiver 2004 is sent to formatter 2006

via bus 2008. The demultiplexing receiver 2004 tells the formatter what virtual link each byte was recovered from in any way. One way is to put each byte (9-bit byte if the 9th bit technique is in use) into a timeslot on bus 2008 corresponding to the virtual link. Another way would be for the receiver to store each byte in an address space in a memory in receiver shared by the formatter 2006 with a tag as to its virtual link. Another way would be for the receiver to maintain one FIFO stack for each virtual link in a memory and for the formatter to pop the top byte off the stack for each virtual link for each frame. Those skilled in the art will appreciate other ways for the demultiplexing receiver 2004 and the formatter 2006 to cooperate so that the formatter knows which bytes came from which virtual links, and any such structure and form of cooperation will suffice to practice this aspect of the invention.

The formatter 2006 functions to reassemble the 48 byte ATM cells from the bytes recovered by the receiver 2004 using the start codes to locate the beginning and end of each ATM cell in the byte stream. Each ATM cell is buffered in a portion of a buffer memory (not shown) dedicated to storing cells from a particular RU. The formatter then regenerates standard ATM cells with 5 byte headers by regenerating the standard headers from the information regarding which virtual link each ATM cell arrived on. The 48 byte cells stored in the buffer memory are used as the payload sections of each standard 53 byte ATM cell. The standard 53 byte ATM cells are then transmitted to SAR 1375 whereupon processing proceeds as previously described to convert the ATM cells back into the AAL5 sequence of Figure 50D for transmission to router 1371.

Although the teachings of the invention have been presented herein in terms of a few preferred and alternative embodiments, those skilled in the art will appreciate numerous modifications, improvement and substitutions that will serve the same functions without departing from the true spirit and scope of the appended claims. All such modifications, improvement and substitutions are intended to be included within the scope of the claims appended hereto.